APPS063-Codensity T408™ HLS Stream with AES-128 Encryption Application Note

V1.0

2023-01-30

Table of Contents

1 Table of Abbreviations

Abbreviation:	Full Form:
NVMe CLI	Non-volatile memory express command line interface
AI	Artificial Intelligence
API	Application Programming Interface
AUD	Access Unit Delimiters
AV	Audio Video
AV1	Alliance Open Media Video 1 Codec
CABAC	Context Adaptive Binary Arithmetic Coding
CBR	Constant Bit Rate
CPU	Central Processing Unit
CRF	Constant Rate Factor
DSP	Digital Signal Processing
EOF	End Of File
FIXQB	Fixed Quantization Parameter
FF	Form Factor
FPS	Frames Per Second
GOP	Group of pictures
HDR	High Dynamic Range
HHHL	Half Height, Half Length
HHFL	Half Height, Full Length
HLG	Hybrid Log Gamma
HRD	Hypothetical Reference Decoder
IE	Inference Engine
Mbps	Mega Bit per second
MBps	Mega Byte per second
NLP	Natural Language Processing
NPU	Neural Processing Unit
NVMe	Non-Volatile Memory express
PCIe	Peripheral Component Interconnect express
PPS	Picture Parameter Set
QP	Quoted Printable
RGB	Red Green Blue
RGBA	Red Green Blue Alpha
RDO	Optimization
ROI	Region of interest

SDK	Software Development Kit
SEI	Supplemental Enhancement Information
SRIOV	Single Root I/O Virtualization
SPS	Sequence Parameter Set
VBR	Variable Bit Rat
VCL	Video Coding Layer
VPS	Video Parameter Set
VUI	Video Usability Information

2 References

Technical Note:

Specifications:

3 Background

This document describes usage of HLS streaming with AES-127 encryption for a T408 encoded output.

3.1 Intended Audience

This document is intended to help engineers/technicians/developers with T408 usage with encrypted ffmpeg encoded outputs.

3.2 Compatibility

Software Compatibility

Release 2.5.0 or newer supports NETINT Codensity T408/T432 Video Transcoder hardware.

Hardware Compatibility

Release 2.5.0 or newer supports NETINT Codensity T408/T432 Video Transcoder hardware.

Operation system

N/A.

4 ffmpeg HLS encrypted stream

The ffmpeg application supports encrypting the encoded output. This section details the ffmpeg use case with streaming a T408 encoded output with HLS AES-128 encryption.

4.1 Pre-Requisite

1] Nginx web server configured and running with http and https support. Setup of nginx is not in the scope of this document.

2] ffmpeg compiled with openssl support. Refer Appendix A.

3] An enc.key and enc.keyinfo file available in /var/www/html folder. Refer Appendix B. It is recommended to share the enc.key over https.

4.2 ffmpeg runtime

Launch a ffmpeg instance with HLS streamed output as per command below. The command is transcoding a H.264 input to a HEVC output streamed over HLS.

```
nvme@CLI168:~$ ffmpeg -hide_banner -y -f concat -c:v
h264_ni_logan_dec -i list.txt -c:v h264_ni_logan_enc -f hls -
hls_key_info_file enc.keyinfo -hls_segment_filename
/var/www/html/hls0/segment%d.ts -hls_time 4 -master_pl_name
master.m3u8 -hls_flags delete_segments
/var/www/html/hls0/master.m3u8
[mov,mp4,m4a,3gp,3g2,mj2 @ 0x55e3c4b7e140] Auto-inserting
h264_mp4toannexb bitstream filter
Input #0, concat, from 'list.txt':
  Duration: N/A, start: 0.000000, bitrate: 4481 kb/s
    Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661),
yuv420p, 1920x1080 [SAR 1:1 DAR 16:9], 4001 kb/s, 60 fps, 60 tbr,
60k tbn, 120 tbc
    Metadata:
      creation_time   : 2013-12-16T17:59:32.000000Z
      handler_name    : GPAC ISO Video Handler
    Stream #0:1(und): Audio: mp3 (mp4a / 0x6134706D), 48000 Hz,
stereo, fltp, 160 kb/s
    Metadata:
      creation_time   : 2013-12-16T17:59:37.000000Z
      handler_name    : GPAC ISO Audio Handler
    Stream #0:2(und): Audio: ac3 (ac-3 / 0x332D6361), 48000 Hz,
5.1(side), fltp, 320 kb/s
```

```
    Metadata:
      creation_time   : 2013-12-16T17:59:37.000000Z
      handler_name    : GPAC ISO Audio Handler
Stream mapping:
  Stream #0:0 -> #0:0 (h264 (h264_ni_logan_dec) -> h264
(h264_ni_logan_enc))
  Stream #0:2 -> #0:1 (ac3 (native) -> aac (native))
Press [q] to stop, [?] for help
[h264_ni_logan_dec @ 0x55e3c4cf7d80] ff_xcoder_dec_receive:
sequence changed: 1920x1080 8bits to 1920x1088 8bits
[h264_ni_logan_enc @ 0x55e3c4cedd80] Session state: 0 allocate
frame fifo.
[h264_ni_logan_enc @ 0x55e3c4cedd80] dts offset: 2
Encoder_complete_info: session_id 0xbf33, total frames input: 0
buffered: 0  completed: 0  output: 0  dropped: 0 ,  inst_errors:
0
Encoder HW[0] INST[127]-average usage:0%
[aac @ 0x55e3c4cf9300] Using a PCE to encode channel layout
"5.1(side)"
[hls @ 0x55e3c4cef680] Opening 'enc.keyinfo' for reading
[hls @ 0x55e3c4cef680] Opening 'enc.key' for reading
Output #0, hls, to '/var/www/html/hls0/master.m3u8':
  Metadata:
    encoder         : Lavf58.45.100
    Stream #0:0(und): Video: h264 (h264_ni_logan_enc),
yuv420p(progressive), 1920x1080 [SAR 1:1 DAR 16:9], q=2-31, 200
kb/s, 60 fps, 90k tbn, 60 tbc
    Metadata:
      creation_time   : 2013-12-16T17:59:32.000000Z
      handler_name    : GPAC ISO Video Handler
      encoder         : Lavc58.91.100 h264_ni_logan_enc
    Stream #0:1(und): Audio: aac (LC), 48000 Hz, 5.1(side), fltp,
394 kb/s
    Metadata:
      creation_time   : 2013-12-16T17:59:37.000000Z
      handler_name    : GPAC ISO Audio Handler
      encoder         : Lavc58.91.100 aac
[hls @ 0x55e3c4cef680] Opening
'crypto:/var/www/html/hls0/segment0.ts' for writingd=2.35x
[hls @ 0x55e3c4cef680] Opening
'/var/www/html/hls0/master.m3u8.tmp' for writing
[hls @ 0x55e3c4cef680] Opening
'crypto:/var/www/html/hls0/segment1.ts' for writingd=2.41x
[hls @ 0x55e3c4cef680] Opening
'/var/www/html/hls0/master.m3u8.tmp' for writing
[hls @ 0x55e3c4cef680] Opening
'crypto:/var/www/html/hls0/segment2.ts' for writing
```

4.3 Playlist file example

The generated playlist example is posted below. The remote player will use the URI for the encryption key to de-encrypt the video segments.

```
nvme@CLI168:/var/www/html/hls0$ cat master.m3u8
#EXTM3U
EXT-X-VERSION:3
#EXT-X-TARGETDURATION:5
#EXT-X-MEDIA-SEQUENCE:211
#EXT-X-KEY:METHOD=AES-128,URI="http://127.0.0.1/enc.key",IV=0xa7f5ff629f81c6314bedfbcd02cc5a81
#EXTINF:4.600000,
segment211.ts
#EXTINF:3.066667,
segment212.ts
#EXTINF:4.600000,
segment213.ts
#EXTINF:3.066667,
segment214.ts
#EXTINF:4.600000,
segment215.ts
nvme@CLI168:/var/www/html/hls0$
```

4.4 ffplay playback runtime

Example of a playback of the encrypted hls stream is posted below.

```
nvme@CLI168:~$ ffplay -hide_banner
http://127.0.0.1/hls0/master.m3u8
[hls @ 0x7f0668000bc0] Skip ('#EXT-X-VERSION:3')
```

```
[hls @ 0x7f0668000bc0] Opening 'http://127.0.0.1/enc.key' for
reading
[hls @ 0x7f0668000bc0] Opening
'crypto+http://127.0.0.1/hls0/segment207.ts' for reading
[hls @ 0x7f0668000bc0] Opening
'crypto+http://127.0.0.1/hls0/segment208.ts' for reading
…
Input #0, hls, from 'http://127.0.0.1/hls0/master.m3u8':
  Duration: N/A, start: 829.364000, bitrate: N/A
  Program 0
    Metadata:
      variant_bitrate : 0
    Stream #0:0: Video: h264 (High) ([27][0][0][0] / 0x001B),
yuv420p(tv), 1920x1080 [SAR 1:1 DAR 16:9], 60 fps, 60 tbr, 90k
tbn, 120 tbc
    Metadata:
      variant_bitrate : 0
    Stream #0:1: Audio: aac (LC) ([15][0][0][0] / 0x000F), 0
channels, fltp
    Metadata:
      variant_bitrate : 0
[hls @ 0x7f0668000bc0] Skip ('#EXT-X-VERSION:3')B sq=    0B f=0/0
[hls @ 0x7f0668000bc0] skipping 1 segments ahead, expired from
playlists
[hls @ 0x7f0668000bc0] Opening
'crypto+http://127.0.0.1/hls0/segment210.ts' for reading
[mpegts @ 0x7f0668007880] Packet corrupt (stream = 0, dts =
75468000).
 839.07 M-V:  0.097 fd= 389 aq=    0KB vq=  232KB sq=    0B f=0/0
nvme@CLI168:~$
```

4.5 Useful Links and Reference

1] https://hlsbook.net/how-to-encrypt-hls-video-with-ffmpeg/
2] https://askubuntu.com/questions/650577/how-to-compile-ffmpeg-with-https-support
3] https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-nginx-in-ubuntu-20-04-1
4] https://www.thesslstore.com/knowledgebase/ssl-install/nginx-ssl-installation/

5 Appendix A – compiling ffmpeg with openssl support

This appendix details the steps needed towards compiling ffmpeg with openssl support.

1] Install openssl package.

```
sudo apt install openssl
sudo apt install libssl-dev
```

2] Compile ffmpeg with openssl.

In the folder where FFmpeg is compiled,

```
cd ~/FFmpeg
make clean
bash build_ffmpeg.sh -dry
```

<copy and paste the output of command above to command line and add - - enable-openssl as shown below>

```
./configure --pkg-config-flags=--static --enable-gpl --enable-
nonfree --extra-ldflags=-lm --extra-ldflags=-ldl --enable-
libxcoder_logan --enable-ni_logan --enable-pthreads --extra-
libs=-lpthread --enable-encoders --enable-decoders --enable-
avfilter --enable-muxers --enable-demuxers --enable-parsers --
enable-x86asm --disable-debug --disable-ffplay --disable-ffprobe
--disable-libx264 --disable-libx265 --disable-cuda-nvcc --
disable-cuda --disable-cuvid --disable-nvdec --disable-nvenc --
disable-libvmaf --enable-static --disable-shared --extra-cflags=-
UXCODER_DLL --extra-cflags=-UNIENC_MULTI_THREAD --extra-cflags=-
UNI_DEC_GSTREAMER_SUPPORT --enable-openssl


make -j
sudo make install
```

Run ‘ffmpeg’ command. It should now show - - enable-openssl as part of the command dump.

```
$ ffmpeg
ffmpeg version 4.3.1 Copyright (c) 2000-2020 the FFmpeg
developers
  built with gcc 9 (Ubuntu 9.4.0-1ubuntu1~20.04.1)
  configuration: --pkg-config-flags=--static --enable-gpl --
enable-nonfree --extra-ldflags=-lm --extra-ldflags=-ldl --enable-
libxcoder_logan --enable-ni_logan --enable-pthreads --extra-
libs=-lpthread --enable-encoders --enable-decoders --enable-
```

```
avfilter --enable-muxers --enable-demuxers --enable-parsers --
enable-x86asm --disable-debug --enable-ffplay --disable-ffprobe -
-disable-libx264 --disable-libx265 --disable-cuda-nvcc --disable-
cuda --disable-cuvid --disable-nvdec --disable-nvenc --disable-
libvmaf --enable-static --disable-shared --extra-cflags=-
UXCODER_DLL --extra-cflags=-UNIENC_MULTI_THREAD --extra-cflags=-
UNI_DEC_GSTREAMER_SUPPORT --enable-openssl
```

6 Appendix B – Sample enc.key and enc.keyinfo file

In the /var/www/html folder run the following steps to generate the enc.key and enc.keyinfo files.

```
cd /var/www
sudo chmod ugo+rwx html
```

```
cd /var/www/html
```

1] Generate enc.key file by running the following command:

```
$ openssl rand 16 > enc.key
```

2] Generate a hex number by running the following command:

```
$ openssl rand -hex 16
```

```
a7f5ff629f81c6314bedfbcd02cc5a81
```

Add this hex number to enc.keyinfo file below.

3] Create enc.keyinfo file with the following contents:

```
http://127.0.0.1/enc.key
```

```
enc.key
```

```
a7f5ff629f81c6314bedfbcd02cc5a81
```

Note – Replace IP address in the above URL based on your webserver IP.

Note – Encrypted key is recommended to be served over https. Configuration of webserver to support https is not in scope of this document.

7 Revision History

Version	Change	Create Date	Author
0.1	Document Created	January 26, 2023	Balwinder Sidhu
1.0	Issued version	January 30, 2023	Balwinder Sidhu

8 Legal Notice

Information in this document is provided in connection with NETINT products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in NETINT's terms and conditions of sale for such products, NETINT assumes no liability whatsoever and NETINT disclaims any express or implied warranty, relating to sale and/or use of NETINT products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right.

A "Mission Critical Application" is any application in which failure of the NETINT Product could result, directly or indirectly, in personal injury or death. Should you purchase or use NETINT's products for any such mission critical application, you shall indemnify and hold NETINT and its subsidiaries, subcontractors and affiliates, and the directors, officers, and employees of each, harmless against all claims costs, damages, and expenses and reasonable attorney's fees arising out of, directly or indirectly, any claim of product liability, personal injury, or death arising in any way out of such mission critical application, whether or not NETINT or its subcontractor was negligent in the design, manufacture, or warning of the NETINT product or any of its parts.

NETINT may make changes to specifications, technical documentation, and product descriptions at any time, without notice. The information here is subject to change without notice. Do not finalize a design with this information. The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications.

NETINT, Codensity, and NETINT Logo are trademarks of NETINT Technologies Inc. All other trademarks or registered trademarks are the property of their respective owners.