Video Access Utilities Manual

9000-62572-15

NMS COMMUNICATIONS

100 Crossing Boulevard
Framingham, MA 01702-5406 USA
www.nmscommunications.com

P/N 9000-62572-15

Revision history

Revision	Release date	Notes
1.0	October 2005	DEH, Video Access 2.0 Beta 2
1.1	December 2005	DEH, Video Access 2.0
1.2	March 2007	EH/LBG, Video Access 3.0 Beta 1
1.3	May 2007	PJP, Video Access 3.0 Beta 2
1.4	July 2007	PJP, Video Access 3.0
1.5	February 2009	DEH, Video Access 3.2
Last modified: January 22, 2009		

Refer to www.nmscommunications.com for product updates and for information about support policies, warranty information, and service offerings.

Table Of Contents

1 Introduction

The *Video Access Utilities Manual* is one in a set of manuals that describe the Video Access product. It describes how to use Video Access utilities to manipulate 3GP files and to monitor 3G-324M calls. These utilities help the application developer troubleshoot the data generated or received by Video Access components.

This manual targets video application developers who use Natural Access and Video Access. It assumes that you are familiar with telephony, switching, Natural Access, and Video Access concepts. If you are not familiar with Video Access, read the *Video Access Overview Manual* to learn about the Video Access concepts and Natural Access features that relate to Video Access before reviewing this manual.

2 Overview of the Video Access Utilities

Video Access utilities overview

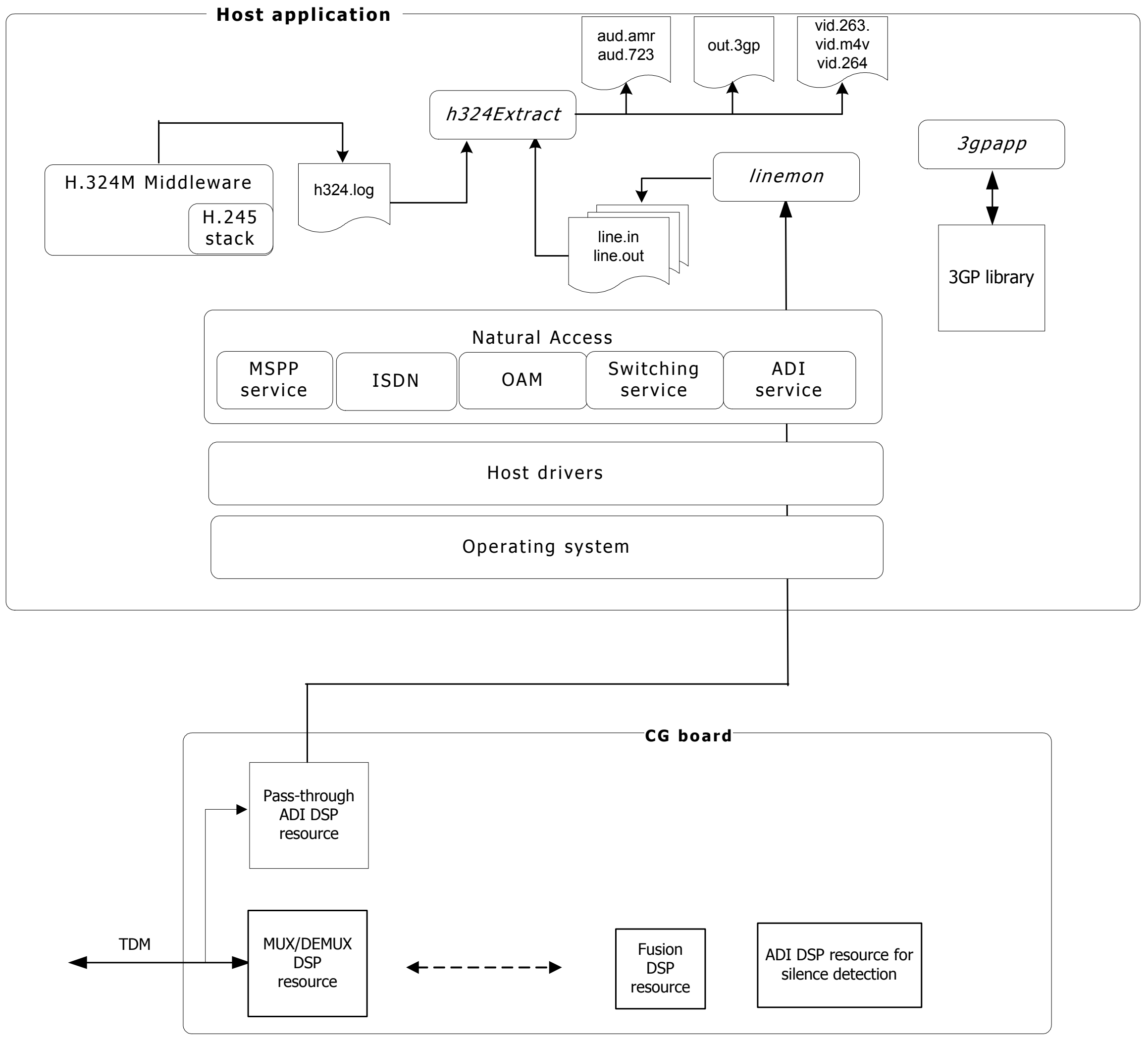
Video Access provides the following utilities:

Utility	Description	For more information, see...
3gpapp	Manipulates 3GP files. With *3gpapp*, you can: • Display information from 3GP files. • Convert NMS-packetized or raw data to 3GP format. • Convert 3GP data to NMS-packetized or raw format. • Rebuild a 3GP file from an existing file, using a different file size or playing duration.	*Overview of 3gpapp* on page 11.
linemon	Records data flowing through a TDM bearer channel. With *linemon,* you can monitor 3G-324M calls and analyze them off-line.	*Overview of linemon* on page 29.
h324Extract	Extracts audio and video data from the recorded TDM bearer channel data. Data can also be used to generate 3GP formatted files.	*Overview of h324Extract* on page 35.

Components used by the Video Access utilities

The following illustration shows the Video Access components used by *3gpapp, linemon,* and *h324Extract*, in the context of a 3G-324M Video Mail configuration:

Note: The names of the inputs and output to *h324Extract* are defaults and may not reflect your installation.



Video Access document set

The following table describes each of the manuals in the Video Access documentation set, along with guidelines for their use:

Manual	Description	Use this manual if...
Video Access Overview Manual	A general introduction to Video Access and its features.	You are new to Video Access. Start with this manual before proceeding to the *Video Mail Application Demonstration Manual.*
Video Mail Application Demonstration Program Manual	How to use *vmsamp*, a functional video mail application built on Video Access and supplied with the product.	You are new to Video Access and want to gain hands-on experience with Video Access technology and code before you start writing your own applications. The *vmsamp* application includes reference code for most of the data structures and API features described in the other Video Access manuals.
3G-324M Interface Developer's Reference Manual	How to use the 3G-324M Interface to connect with 3G-324M terminals capable of audio and video. This manual also describes the 3G-324M Interface capabilities and functions.	You are developing gateway functionality based on the 3G-324M Interface.
Video Messaging Server Interface Developer's Reference Manual	How to play and record audio and video RTP media, and how to use the Video Messaging Server Interface.	Your application will use the Video Messaging Server Interface to process video and audio streams.
Video Access Utilities Manual	How to use the Video Access utilities that are available for manipulating 3GP files and monitoring 3G-324M calls.	You are responsible for Video Access content capture and analysis, or for the manipulation or troubleshooting of data generated or received by Video Access components. The utilities documented here can also be used to manipulate content created outside of Video Access.

Note: For an additional layer of detail about Video Access structures, refer to the Video Access header files.

3 3gpapp

Overview of 3gpapp

3gpapp is an offline utility that manipulates 3GP files. Use *3gpapp* to:

- Display information from 3GP files.
- Convert NMS packetized data or raw data to 3GP format.
- Convert 3GP data to NMS packetized or raw format.
- Create a new 3GP file from an existing 3GP file (called rebuilding a file). The new file can have a different file size or playing duration from the original file.
- Perform skew correction while converting NMS packetized data or raw data to 3GP format or while creating a new 3GP file from an existing 3GP file.
- Create a sync point table while converting NMS packetized data or raw data to 3GP format or while creating a new 3GP file from an existing 3GP file.
- Seek a specific time in a 3GP file while converting 3GP data to NMS packetized or raw format.
- Create hint tracks while converting NMS packetized data to 3GP format or creating a new 3GP file from an existing 3GP file.
- Use hint tracks while converting 3GP data to NMS packetized data.

3gpapp handles three types of input and output formats:

- Raw format
- NMS packetized format
- 3GP format

Raw format

Raw format can be any of the following types of bit streams:

- H.263 baseline profile 0, level 10 - 30 (QCIF, CIF) video elementary bit stream. Video bit streams must conform to ITU-T Recommendation H.263 Annex X as defined in ITU-T Recommendation H.263 1998 and 2000 and 3GPP specifications TS.26.111, TS.26.911, TS.26.140 (H.263).
- H.264 baseline profile level 1 and 2 (QCIF, CIF) video elementary bit stream formatted in the NMS-packetized proprietary format.

  Video bit streams must conform to ITU-T Recommendation H.264 and 3GPP specifications TS.26.111, TS.26.911, TS.26.140 (H.264).

  Note: Raw video to 3GP format conversion is not supported for the H.264 video codec.

- MPEG-4 simple profile level 0 - 3 (QCIF, CIF) video elementary bit stream. *3gpapp* expects an ISO/IEC 14496 simple profile level 0 - 3 bit stream. Time stamping and headers, in particular VOS/VO/VOL headers, should comply with ISO/IEC 14496 and 3GPP TS 26.110 /TS 26.111. This information is needed in the correct format to allow the computation of parameters for 3GP file formatting.
- AMR Narrow-Band IF2 elementary audio bit stream. Audio bit streams must conform to ETSI/AMR, Adaptive Multi-Rate speech codec (AMR), in IF2 framing format, in any of the eight AMR compressed data rates.

NMS packetized format

NMS packetized format can be any of the previously described raw bit streams, formatted in the NMS packetized proprietary format. For more information about this format, refer to the *Video Messaging Server Interface Developer's Reference Manual*.

The following table shows the NMS convention for NMS packetized file name extensions generated automatically by *3gpapp*:

Extension	Description
.263	Elementary H.263 bit stream
.n263n	NMS packetized H.263 packetized with RFC 2190
.n263	NMS packetized H.263 packetized with RFC 2429
.264	Elementary H.264 bit stream
.n264	NMS-packetized H.264 packetized with RFC 3984
.m4v	Elementary MPEG-4 bit stream
.nm4v	NMS packetized MPEG-4 packetized with RFC 3016
.amr	AMR raw bit stream
.namr	NMS packetized AMR IF2 packetized with RFC 3267

3GP format

The 3GP file format is compliant with the 3GPP TS 26.244 specification and conforms to 3GP file format Basic Profile. A 3GP file contains one video track (either H.263 baseline level 10-30 bit stream, H.264 baseline profile level 1-2 bit stream or MPEG-4 simple profile level 0-3 bit stream), one audio track (IF1 AMR NB audio elementary bit stream), or both.

3gpapp syntax

3gpapp is installed in the *C:\nms\bin\* directory in Windows or the */opt/nms/bin/* directory in UNIX.

3gpapp uses the following syntax:

```
3gpapp command command-option [file1] [file2] [file3]
```

where:

Argument	Description
command	The command to execute. Valid values are described in the command table.
command-option	The command option associated with a ***command***. Valid values are described in the command-option table.
file1, ***file2***, ***file3***	Full names (including extensions) of the files being acted on by a *3gpapp* command. For more information, see the individual command descriptions.

The following table describes the *3gpapp* commands and lists their associated options:

Command	Description	Associated command options
-c	Create a 3GP file (***file1***) from an existing audio file (***file2***), video file (***file3***), or both. Audio and video files can be either in NMS packetized or raw format.	-b: Buffer size -H: Header check -I: Interleaving depth -g: Hint tracks -s: Maximum size -t: Maximum time -v: Verbosity level -w: Creation mode -y: Sync points -z: Skew correction
-d	Display detailed information from a 3GP file.	-v: Verbosity level
-e	Extract audio data, video data, or both from a 3gp file (***file1***), and create raw or NMS packetized files (***file2***). The file extension for ***file2*** is a codec-dependent extension.	-b: Buffer size -f: Format of extracted media data -m: NMS packet max transfer unit -Q: Hint tracks -R: H.263 packetization format -v: Verbosity level -Y: Seek
-h	Display a help screen and exit.	NA
-i	Display general information from an existing 3GP file (***file1***).	-v: Verbosity level

Command	Description	Associated command options
-r	Rebuild a 3GP file. This process creates a new 3GP file (***file2***) with media data extracted from an existing 3GP file (***file1***).	-b: Buffer size -c: Creation mode -H: Header check -I: Interleaving depth -q: Hint tracks -s: Size -t: Maximum time -v: Verbosity level -y: Sync points -z: Skew correction

The following table describes the *3gpapp* command options:

Option	Description	Use with these commands
-b***size***	Buffer size to use, in bytes. Default is 0 (use the whole stream).	-c, -e, and -r
-f***format***	Format of media data. Valid values are: • 0 (default): NMS packetized • 1: Raw	-e Raw format cannot be used with the partial buffer option (-b with a non-zero value).
-H	NMS header check when building the 3GP file.	-c and -r
-I***depth***	Interleaving depth of streams, in milliseconds. The default is 1000.	-c and -r
-m***n***	NMS packet max transfer unit (mtu) size, in bytes. The default is 1432.	-e
-p	Read SDP information from a text file and insert it into a 3GP file.	-c and -r
-P	Extract SDP information from a 3GP file and write it to a text file.	-e
-q	Add RTP hint tracks to the 3GP file.	-c and -r
-Q	Use RTP hint tracks to packetize media data.	-e
-R	Changes the packetization mode for H.263 media to RFC 2190, when specified. The default packetization mode is RFC 2429.	-e
-s***size***	Maximum size of the 3GP file, in bytes.	-c and -r
-t***time***	Maximum file duration, in milliseconds, for all included media streams in the 3GP file. A value of 0 (zero) indicates that the time is not limited.	-c and -r
-w***mode***	Method of creating the 3GP file. Valid values are: • 1: Write directly to the 3GP file. • 2 (default): Write to temporary files.	-c and -r

Option	Description	Use with these commands
-v***level***	Verbosity level for standard output. Valid values are: • 0: None • 1: Complete information, including debug, warning and error information • 2: Debug information • 4: Warning information • 8: Error information	All
-y	Create sync points when building the 3GP file.	-c and -r
-Y***seektime***	Seek to a specific time in milliseconds.	-e
-z***skewtime***	Enable skew correction by insertion and removal of audio frames.	-c and -r

Examples: Displaying information from 3GP files

The following examples show how to display both general information and detailed information from 3GP files:

- Displaying general information from a 3GP file
- Displaying detailed information from a 3GP file

Displaying general information from a 3GP file

This example displays general information from the *sample.3gp* file.

Command

```
3gpapp -i sample.3gp
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725
GETTING FILE INFO FROM sample.3gp
        Descriptor:
          file type                    = 1 (3GP)
          format                       = 3gp5
          version                      = 512 (v5.2.0)
          Stream 0
          stream type                  = 1 (AUDIO)
          stream id                    = 1
          stream codec                 = 1 (AMR)
          duration                     = 15980 ms
          stream data size             = 25568 bytes
          stream data rate             = 12.80 kbits/second
          sample avg rate              = 50.00 frames/second
          Stream 1:
          stream type                  = 2 (VIDEO)
          stream id                    = 2
          stream codec                 = 3 (H263)
          duration                     = 16116 ms
          stream data size             = 70723 bytes
          stream data rate             = 35.11 kbits/second
          sample avg rate              = 9.68 frames/second
```

Displaying detailed information from a 3GP file

This example displays detailed information from the *sample.3gp* file.

Command

```
3gpapp -d sample.3gp
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725
GETTING FILE INFO FROM sample.3gp
        Descriptor:
          file type                     = 1 (3GP)
          format                        = 3gp5
          version                       = 512 (v5.2.0)
-> detailed file information:
        Presentation:
          blk size                      = 28 (typ.= 28)
          creation time                 = 3176198412
          duration                      = 16114 ms
          stream count                  = 2
          streams                       = 0
          [flags - reserved]            = 0x00000000
          [max bitrate - reserved]      = 0
          max interleave                = 1101 ms
         ----------------------------------------------
         Stream:
          blk size                      = 228
          * Header size                   = 64 (typ.= 64)
          stream type                   = 1 (AUDIO)
          codec                         = 1 (AMR)
          stream ID                     = 1
          creation time                 = Wed Aug 25 09:20:12 2004
          [alternate group ID - rsvd]   = 0
          [switching group ID - rsvd]   = 0
          [select attributes - rsvd]    = 0
          duration                      = 15980 ms
          timescale                     = 1000 Hz
          media handler Name            = soun
          sample size                   = 32 bytes
          sample count                  = 799
          max sample size               = 0 (not available)
          byte count                    = 25568
          [flags - reserved]            = 0x00000000
          [start delay - reserved]      = 0
          => Stream data size           = 25568 bytes
          => Stream data rate           = 12.80 kbits/second
          => Sample avg rate            = 50.00 frames/second
          * Audio format size           = 12 (typ.= 12)
          channel count                 = 2
          sample size                   = 16 bits
          [flags - reserved]            = 0x00000000
          * AMR mode set                = 0x0080 - 12.2 kbits/s
          mode chg period               = 0
          frames / sample               = 1
          vendor                        = VXYZ
         ----------------------------------------------
        Stream:
          blk size                      = 232
          * Header size                 = 64 (typ.= 64)
          stream type                   = 2 (VIDEO)
          codec                         = 3 (H263)
          stream ID                     = 2
          creation time                 = Wed Aug 25 09:20:12 2004
          [alternate group ID - rsvd]   = 0
          [switching group ID - rsvd]   = 0
          [select attributes - rsvd]    = 0
          duration                      = 16116 ms
          timescale                     = 90000 Hz
          media handler Name            = vide
          sample size                  = 0 (variable)
          sample count                 = 156
          max sample size              = 0 (not available)
          byte count                   = 70723
          [flags - reserved]           = 0x00000000
          [start delay - reserved]     = 0
          => Stream data size          = 70723 bytes
          => Stream data rate          = 35.11 kbits/second
          => Sample avg rate           = 9.68 frames/second
          * Video format size          = 16 (typ.= 16)
          width                        = 176 pixels
          height                       = 144 pixels
          layer                        = 0
          horiz resolution             = 72 dpi
          vert resolution              = 72 dpi
          [flags - reserved]           = 0x00000000
          * H263 level                 = 10
          H263 profile                 = 0
          [avg bitrate - reserved]     = 0
          [max bitrate - reserved]     = 0
          [vendor - reserved]          = VXYZ
          => Movie data size           = 94.03 kbytes
          => Movie data rate           = 5.84 kbytes/second
```

Examples: Rebuilding 3GP files

The following examples show how to rebuild 3GP files:

- Rebuilding a 3GP file using a playing duration limit
- Rebuilding a 3GP file using a size limit

Rebuilding a 3GP file using a playing duration limit

The following example shows how to rebuild the *sample.3gp* file using a playing duration of 10 seconds (10,000 milliseconds). The example puts data in a new 3GP file called *sample_T10000.3gp.* The data in this file can take up to 10 seconds to replay.

Command

```
3gpapp -r -t10000 sample.3gp sample_T10000.3gp
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725
REBUILDING sample.3gp INTO sample_t10000.3gp WITH AUDIO AND VIDEO
        File sample.3gp rebuilt into file sample_t10000.3gp for AUDIO AND VIDEO

GETTING FILE INFO FROM sample.3gp
        Descriptor:
          file type                      = 1 (3GP)
          format                         = 3gp5
          version                        = 512 (v5.2.0)

        Stream 0:
          stream type                    = 1 (AUDIO)
          stream id                      = 1
          stream codec                   = 1 (AMR)
          duration                       = 15980 ms
          stream data size               = 25568 bytes
          stream data rate               = 12.80 kbits/second
          sample avg rate                = 50.00 frames/second

        Stream 1:
          stream type                    = 2 (VIDEO)
          stream id                      = 2
          stream codec                   = 3 (H263)
          duration                       = 16116 ms
          stream data size               = 70723 bytes
          stream data rate               = 35.11 kbits/second
          sample avg rate                = 9.68 frames/second

GETTING FILE INFO FROM sample_t10000.3gp
        Descriptor:
          file type                      = 1 (3GP)
          format                         = 3gp5
          version                        = 512 (v5.2.0)

        Stream 0:
          stream type                    = 1 (AUDIO)
          stream id                      = 1
          stream codec                   = 1 (AMR)
          duration                       = 10000 ms
          stream data size               = 16000 bytes
          stream data rate               = 12.80 kbits/second
          sample avg rate                = 50.00 frames/second

        Stream 1:
          stream type                    = 2 (VIDEO)
          stream id                      = 2
          stream codec                   = 3 (H263)
          duration                       = 9976 ms
          stream data size               = 43542 bytes
          stream data rate               = 34.92 kbits/second
          sample avg rate                = 10.02 frames/second
```

Rebuilding a 3GP file using a size limit

The following example shows how to rebuild the *sample.3gp* file using a size limit of 50000 bytes. The example puts data in a new 3GP file called *sample_S50000.3gp.* It limits the size of this file to 50000 bytes.

Command

```
3gpapp -r -s50000 sample.3gp sample_S50000.3gp
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725
REBUILDING sample.3gp INTO sample_S50000.3gp WITH AUDIO AND VIDEO
        File sample.3gp rebuilt into file sample_S50000.3gp for AUDIO AND VIDEO

GETTING FILE INFO FROM sample.3gp
        Descriptor:
          file type                       = 1 (3GP)
          format                          = 3gp5
          version                         = 512 (v5.2.0)

        Stream 0:
          stream type                     = 1 (AUDIO)
          stream id                       = 1
          stream codec                    = 1 (AMR)
          duration                        = 15980 ms
          stream data size                = 25568 bytes
          stream data rate                = 12.80 kbits/second
          sample avg rate                 = 50.00 frames/second

        Stream 1:
          stream type                     = 2 (VIDEO)
          stream id                       = 2
          stream codec                    = 3 (H263)
          duration                        = 16116 ms
          stream data size                = 70723 bytes
          stream data rate                = 35.11 kbits/second
          sample avg rate                 = 9.68 frames/second

GETTING FILE INFO FROM sample_S50000.3gp
        Descriptor:
          file type                       = 1 (3GP)
          format                          = 3GP6
          version                         = 256 (v6.1.0)

        Stream 0:
          stream type                     = 1 (AUDIO)
          stream id                       = 1
          stream codec                    = 1 (AMR)
          duration                        = 7980 ms
          stream data size                = 12768 bytes
          stream data rate                = 12.80 kbits/second
          sample avg rate                 = 50.00 frames/second

         Stream 1:
          stream type                     = 2 (VIDEO)
          stream id                       = 2
          stream codec                    = 3 (H263)
          duration                        = 7707 ms
          stream data size                = 33324 bytes
          stream data rate                = 34.59 kbits/second
          sample avg rate                 =  9.99 frames/second
```

Examples: Converting 3GP data to other formats

The following examples show how to convert 3GP data to other formats:

- Converting 3GP data to RFC 2429 NMS packetized format
- Converting 3GP data to RFC 2190 NMS packetized format
- Converting 3GP data to raw format

Converting 3GP data to RFC 2429 NMS packetized format

The following example converts data in a 3GP file named *sample.3gp* to NMS packetized data in accordance with the RFC 2429 packetization mode. The example creates two files in NMS packetized format:

- *sample.namr* contains the audio data from the 3GP file.
- *sample.n263* contains the video data from the 3GP file.

Command

```
3gpapp -e sample.3gp sample
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725

EXTRACT from sample.3gp INTO sample.xxx FOR AUDIO AND VIDEO

34357 bytes copied to file: sample.namr

Total: 34357 bytes copied to file: sample.namr

72595 bytes copied to file: sample.n263

Total: 72595 bytes copied to file: sample.n263
```

Converting 3GP data to RFC 2190 NMS packetized format

The following example converts a 3GP file named *sample.3gp* to NMS packetized format in accordance with RFC 2190. It creates two files in NMS packetized format:

- *sample.namr* contains the audio data from the 3GP file.
- *sample.n263n* contains the video data from the 3GP file.

Command

```
3gpapp -e –R sample.3gp sample
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725

EXTRACT from sample.3gp INTO sample.xxx FOR AUDIO AND VIDEO

34357 bytes copied to file: sample.namr

Total: 34357 bytes copied to file: sample.namr

73219 bytes copied to file: sample.n263n

Total: 73219 bytes copied to file: sample.n263n
```

Converting 3GP data to raw format

The following example converts a 3GP file named *sample.3gp* to raw data. It creates two files in raw format:

- *sample.amr* contains the audio data from the 3GP file.
- *sample.263* contains the video data from the 3GP file.

Command

```
3gpapp -e -f1 sample.3gp sample
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725

EXTRACT from sample.3gp INTO sample.xxx FOR AUDIO AND VIDEO

24769 bytes copied to file: sample.amr

Total: 24769 bytes copied to file: sample.amr

70723 bytes copied to file: sample.263

Total: 70723 bytes copied to file: sample.263
```

Examples: Converting non-3GP data to 3GP format

The following examples show how to convert non-3GP data to 3GP format:

- Converting RFC 2429 NMS packetized data to 3GP format
- Converting RFC 2190 NMS packetized data to 3GP format
- Converting raw data to 3GP format

Converting RFC 2429 NMS packetized data to 3GP format

The following example converts an audio file (*sample.namr*) and a video file (*sample.n263*) in RFC 2429 NMS packetized format to 3GP format. It creates a new 3GP file named *newsample.3gp* that contains both audio and video data.

Command

```
3gpapp -c newsample.3gp sample.namr sample.n263
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725
CREATING newsample.3gp FROM sample.namr AND sample.n263
Reading data from file: sample.namr
BufferSize=34357, total read 34357 bytes
34357 bytes read in file: sample.namr
25568 bytes written in from file sample.namr to file: newsample.3gp
25568 bytes written in from file sample.namr to file: newsample.3gp
Reading data from file: sample.n263
BufferSize=72595, total read 72595 bytes
72595 bytes read in file: sample.n263
70723 bytes written in from file sample.n263 to file: newsample.3gp
70723 bytes written in from file sample.n263 to file: newsample.3gp
```

Converting RFC 2190 NMS packetized data to 3GP format

The following example converts an audio file (*sample.namr*) and a video file (*sample.n263n*) in RFC 2190 NMS packetized format to 3GP format. It creates a new 3GP file named *newsample.3gp* that contains both audio and video data.

Command

```
3gpapp -R -c newsample.3gp sample.namr sample.n263n
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725
CREATING newsample.3gp FROM sample.namr AND sample.n263n
Reading data from file: sample.namr
BufferSize=34357, total read 34357 bytes
34357 bytes read in file: sample.namr
25568 bytes written in from file sample.namr to file: newsample.3gp
25568 bytes written in from file sample.namr to file: newsample.3gp
Reading data from file: sample.n263n
BufferSize=73219, total read 73219 bytes
73219 bytes read in file: sample.n263n
70723 bytes written in from file sample.n263n to file: newsample.3gp
70723 bytes written in from file sample.n263n to file: newsample.3gp
```

Converting raw data to 3GP format

The following example converts an audio file (*sample.amr*) and a video file (*sample.263*) in raw format to 3GP format. It creates a new 3GP file called *newsample.3gp* that contains both audio and video data.

Command

```
3gpapp -f1 -c newsample.3gp sample.amr sample.263
```

Returned data

```
3gpapp - NMS Communications 2005 - v1.0.725
CREATING newsample.3gp FROM sample.amr AND sample.263
Reading data from file: sample.amr
24769 bytes read in file: sample.amr
25568 bytes written in from file sample.amr to file: newsample.3gp
25568 bytes written in from file sample.amr to file: newsample.3gp
Reading data from file: sample.263
70723 bytes read in file: sample.263
70723 bytes written in from file sample.263 to file: newsample.3gp
70723 bytes written in from file sample.263 to file: newsample.3gp
```

Examples: Performing skew correction

Skew correction allows the insertion or removal of audio frames to/from an audio stream. The following examples show how to perform skew correction when building a 3GP file:

- Performing skew correction by inserting audio frames into a 3GP file
- Performing skew correction by removing audio frames from a 3GP file

Performing skew correction by inserting audio frames into a 3GP file

The following example performs skew correction by creating the *newsample.3gp* file. The skew correction value is 100 ms. The first audio frame is inserted five times into the start of an audio stream.

Computation of frames inserted:

Number of frames inserted = 100 ms/20 ms = 5

where the skew correction = 100 ms and the audio frame duration (AMR) = 20 ms.

Command

```
3gpapp -z100 -c newsample.3gp sample.namr sample.n263
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
Skew correction is set as 100
CREATING newsample.3gp FROM sample.namr AND sample.n263
Reading data from file: sample.namr
BufferSize=91117, total read 91117 bytes
91117 bytes read in file: sample.namr
67968 bytes written in from file sample.namr to file: newsample.3gp
67968 bytes written in from file sample.namr to file: newsample.3gp
Reading data from file: sample.n263
BufferSize=234549, total read 234549 bytes
234549 bytes read in file: sample.n263
231429 bytes written in from file sample.n263 to file: newsample.3gp
231429 bytes written in from file sample.n263 to file: newsample.3gp
```

Performing skew correction by removing audio frames from a 3GP file

The following example performs skew correction by creating the *newsample.3gp* file. The skew correction value is -100 ms. The first five audio frames are removed from the audio stream. For example, the first five audio frames are not inserted into the audio frames, they are skipped.

Command

```
3gpapp -z-100 -c newsample.3gp sample.namr sample.n263
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
Skew correction is set as -100
CREATING newsample.3gp FROM sample.namr AND sample.n263
Reading data from file: sample.namr
BufferSize=91117, total read 91117 bytes
91117 bytes read in file: sample.namr
67648 bytes written in from file sample.namr to file: newsample.3gp
67648 bytes written in from file sample.namr to file: newsample.3gp
Reading data from file: sample.n263
BufferSize=234549, total read 234549 bytes
234549 bytes read in file: sample.n263
231429 bytes written in from file sample.n263 to file: newsample.3gp
231429 bytes written in from file sample.n263 to file: newsample.3gp
```

Examples: Using random access capabilities

The following examples show how to use random access capabilities:

- Creating sync points when building a 3GP file
- Printing a sync point table for a 3GP file
- Seeking to a specific time in a 3GP file

Creating sync points when building a 3GP file

The following example shows how to create sync points for a 3GP file. A sync point will be added for every video sync frame (I-frame).

Command

```
3gpapp -y -c newsample.3gp sample.namr sample.n263
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
Write sync points
CREATING newsample.3gp FROM sample.namr AND sample.n263
Reading data from file: sample.namr
BufferSize=91117, total read 91117 bytes
91117 bytes read in file: sample.namr
67808 bytes written in from file sample.namr to file: newsample.3gp
67808 bytes written in from file sample.namr to file: newsample.3gp
Reading data from file: sample.n263
BufferSize=234549, total read 234549 bytes
234549 bytes read in file: sample.n263
231429 bytes written in from file sample.n263 to file: newsample.3gp
231429 bytes written in from file sample.n263 to file: newsample.3gp
```

Printing a sync point table for a 3GP file

The following example shows how to print a sync point table for a 3GP file.

Command

```
3gpapp –i newsample.3gp (or use –d option)
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
GETTING FILE INFO FROM newsample.3gp
       Descriptor:
         file type                    = 1 (3GP)
         format                       = 3GP6
         version                      = 256 (v6.1.0)
       Stream 0:
         stream type                  = 1 (AUDIO)
         stream id                    = 1
         stream codec                 = 1 (AMR)
         duration                     = 42380 ms
         stream data size             = 67808 bytes
         stream data rate             = 12.80 kbits/second
         sample avg rate              = 50.00 frames/second
       Stream 1:
         stream type                  = 2 (VIDEO)
         stream id                    = 2
         stream codec                 = 3 (H263)
         duration                     = 42417 ms
         stream data size             = 231429 bytes
         stream data rate             = 43.65 kbits/second
         sample avg rate              = 5.21 frames/second
       Sync table information:
         Sync table size = 10
         Sync      Sync
         Point     Time(ms)
         ------------------
             0            0
             1         5000
             2        10000
             3        15041
             4        20041
             5        22208
             6        27208
             7        32375
             8        37416
             9        40750
```

Seeking to a specific time in a 3GP file

The following example shows how to seek to a time in a 3GP file. This example seeks to the time T = 20 seconds (20000 ms). A seek operation can be done once and only once, at the start of the file.

Command

```
3gpapp –Y20000 newsample.3gp
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
Seeking ahead n=20000 msecs.
EXTRACT from newsample.3gp INTO newsample.xxx FOR AUDIO AND VIDEO
48031 bytes copied to file: newsample.namr
Total: 1358523 bytes copied to file: newsample.namr
121450 bytes copied to file: newsample.n263
Total: 121450 bytes copied to file: newsample.n263
```

Examples: Using SDP capabilities

The SDP feature allows an application to store and retrieve SDP information. The following examples show how to use SDP capabilities:

- Creating a 3GP file with SDP information
- Extracting SDP information from a 3GP file

Creating a 3GP file with SDP information

The following example shows how to create a 3GP file with SDP information.

Command

```
3gpapp -p in.sdp -c newsample.3gp sample.namr sample.n263
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
CREATING newsample.3gp FROM sample.namr AND sample.n263
Reading data from file: sample.namr
BufferSize=91117, total read 91117 bytes
91117 bytes read in file: sample.namr
67808 bytes written in from file sample.namr to file: newsample.3gp
67808 bytes written in from file sample.namr to file: newsample.3gp
Reading data from file: sample.n263
BufferSize=234549, total read 234549 bytes
234549 bytes read in file: sample.n263
231429 bytes written in from file sample.n263 to file: newsample.3gp
231429 bytes written in from file sample.n263 to file: newsample.3gp
```

Extracting SDP information from a 3GP file

The following example shows how to extract SDP information from a 3GP file.

Command

```
3gpapp -P out.sdp -e newsample.3gp newsample
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
EXTRACT from newsample.3gp INTO newsample.xxx FOR AUDIO AND VIDEO
91117 bytes copied to file: newsample.namr
Total: 1401609 bytes copied to file: newsample.namr
234549 bytes copied to file: newsample.n263
Total: 234549 bytes copied to file: newsample.n263
```

The user may check the SDP information in the file out.sdp.

Examples: Using hint track capabilities

The hint tracks feature allows a streaming server to create RTP streams from a 3GP file without requiring the server to know anything about the media type, compression, or payload format. The following examples show how to use hint tracks capabilities:

- Creating a 3GP file with hint tracks
- Using hint tracks from a 3GP file to packetize media data

Creating a 3GP file with hint tracks

The following example shows how to create a 3GP file with hint tracks. A hint track will be added for each media stream.

Command

```
3gpapp -q -c newsample.3gp sample.namr sample.n263
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
Adding RTP hint track.
CREATING newsample.3gp FROM sample.namr AND sample.n263
Reading data from file: sample.namr
BufferSize=91117, total read 91117 bytes
91117 bytes read in file: sample.namr
67808 bytes written in from file sample.namr to file: newsample.3gp
67808 bytes written in from file sample.namr to file: newsample.3gp
Reading data from file: sample.n263
BufferSize=234549, total read 234549 bytes
234549 bytes read in file: sample.n263
231429 bytes written in from file sample.n263 to file: newsample.3gp
231429 bytes written in from file sample.n263 to file: newsample.3gp
```

Using hint tracks from a 3GP file to packetize media data

The following example shows how to use hint tracks from a 3GP file to packetize media data.

Command

```
3gpapp -Q -e newsample.3gp newsample
```

Returned data

```
3gpapp - NMS Communications 2007 - v1.0.725
Using RTP hint track.
EXTRACT from newsample.3gp INTO newsample.xxx FOR AUDIO AND VIDEO
91117 bytes copied to file: newsample.namr
Total: 1401609 bytes copied to file: newsample.namr
234549 bytes copied to file: newsample.n263
Total: 234549 bytes copied to file: newsample.n263
```

4 linemon

Overview of linemon

The *linemon* utility records data flowing through a TDM bearer channel. Use *linemon* to prepare data for off-line analysis. To analyze the data, use *h324Extract.* For more information, refer to *Overview of h324Extract* on page 35.

Note: The bearer channel to be monitored must originate from the CG trunk interface, not from the H100/H110 bus.

How linemon works

linemon requires a DSP pass-through recording resource on the CG board. It works as follows:

- Switches the inbound TDM trunk-bearer channel to be recorded to a DSP pass-through recording resource.
- Switches the outbound TDM bearer channel to be recorded to a second DSP pass-through recording resource.
- Captures the raw 64 kbps data flowing in both directions of a 3G-324M call.
- Stores the captured data on the hard drive of the host application in two files: *filename.in* and *filename.out*. By default, the filename prefix is *line* (as in *line.in*). You can change the default prefix when you run *linemon*, as described in *linemon syntax* on page 32.

linemon works simultaneously with the Video Access host application that drives the 3G-324M interface.

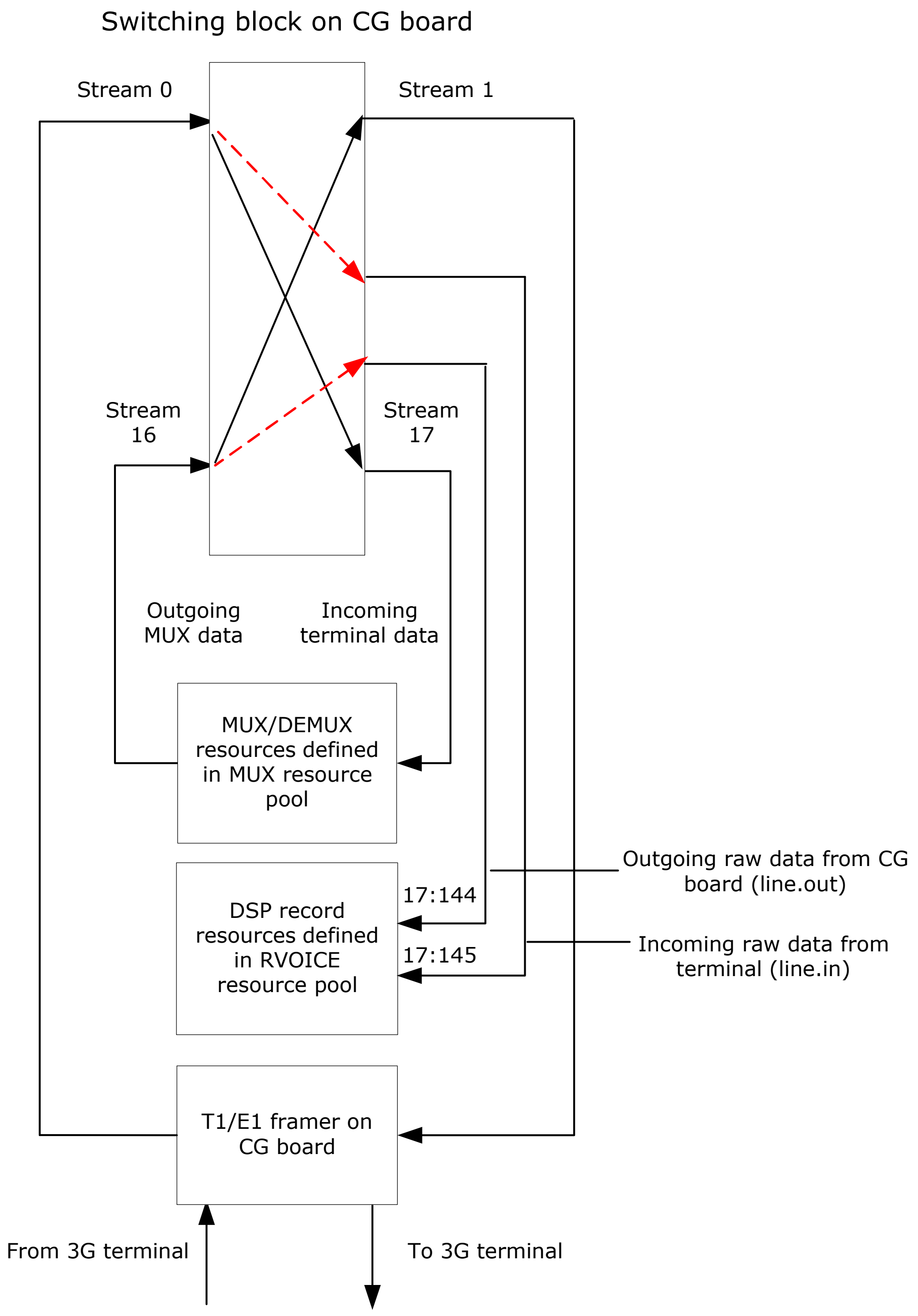
linemon switching

linemon monitors two timeslots:

- A specific timeslot (receive direction) on a T1 or E1 trunk of the CG board, as specified by –l option.
- A specific DSP MUX timeslot (transmit direction), as specified by the –n option.

If default switching is used on the CG board, and if the MUX DSP resource starts on DSP timeslot 0, then both timeslots are the same, and the –n option can be ignored.

The following illustration shows an example of switching performed by *linemon* to monitor a 3G-324M channel on the first E1 trunk of the CG board. The switching is shown with dotted lines.


Switching block on CG board
Stream 0
Stream 1
Stream 16
Stream 17
Outgoing MUX data
Incoming terminal data
MUX/DEMUX resources defined in MUX resource pool
DSP record resources defined in RVOICE resource pool
17:144
17:145
Outgoing raw data from CG board (line.out)
Incoming raw data from terminal (line.in)
T1/E1 framer on CG board
From 3G terminal
To 3G terminal

Setting up linemon

To set up *linemon*, follow these steps:

Step	Action

1	Use the **dbg_nl_passthru_rec** function to configure the CG board with a DSP pool of resources loaded with rvoice DPF. Reserve two timeslots in this pool, as shown in the following example:
	`Resource[1].Name = RSC1`
	`Resource[1].Size = 2`
	`Resource[1].TCPs = nocc`
	`Resource[1].DSPs = 31`
	`Resource[1].StartTimeSlot = 144`
	`Resource[1].Definitions = (rvoice.dbg_nl_passthru_rec & dtmf.det_all)`
	Note: DSP timeslot 144 is the default timeslot used by *linemon.* You can change this timeslot in the *linemon* command line.

2	Set up the Debug Rvoice DSP in NO_LAW so the DSP will not compand: `DSP.C5x[31].XLaw = NO_LAW`

For information about configuring the board, see the installation and developer's manual for the board.

Recording a call with linemon

To record a complete call with *linemon*, follow these steps:

Step	Action
1	Know which channel you want to monitor.
2	Start *linemon* on the channel before 3G-324M call setup.
3	Stop *linemon* after 3G-324M call tear-down.

For information about *linemon* commands, see *linemon syntax* on page 32.

linemon syntax

linemon is installed in the *C:\nms\bin\* directory in Windows or the */opt/nms/bin/ directory* in UNIX.

linemon uses the following syntax:

```
linemon argument [argument]...
```

where:

Argument	Description
-b ***board***	Specifies the board number to record. The default board number is 0.
-d	Display recorded data to the screen.
-e	Uses E1 mode, which assumes each trunk has 30 timeslots. If neither -e nor -E is specified, *linemon* uses T1 mode.
-E	Uses E1 raw mode, which assumes each trunk has 31 timeslots.
-f ***filename***	Specifies a prefix for the record files. The default prefix is *line*.
-h	Prints a Help menu.
-l ***timeslot***	Specifies a timeslot on the T1 or E1 trunk to be monitored. The default timeslot is 0.
-n ***timeslot***	Specifies the DSP MUX timeslot to which the T1 or E1 trunk is connected. The default timeslot is 0.
-m ***timeslot***	Specifies the DSP timeslot to use for monitoring. The default timeslot is 144.

linemon examples

The following examples show how to use *linemon* to record and display specific trunk timeslots:

- Recording and displaying trunk timeslot 0 in T1 mode
- Recording trunk timeslot 40 in E1 mode
- Recording timeslot 0 in T1 mode using a specific board and specific monitors

For all *linemon* commands, the -n option can be ignored when both of the following are true:

- Default switching between trunk timeslots and MUX DSP timeslots is used on the CG board.
- The MUX DSP resource starts on DSP timeslot 0.

Recording and displaying trunk timeslot 0 in T1 mode

The following example records and displays trunk timeslot 0 using T1 mode (the default mode). The command shown here also works for an E1 trunk, if the monitored timeslot is less than 24.

Command

```
linemon -d
```

Received data

```
Display Output
Monitoring timeslot 0 using DSP timeslots 144 (line.out) and 145 (line.in)
Making switch connections 16:0->17:144 (line.out) and 0:0->17:145 (line.in).
Recording of Line Out and Line In Started.
       0    0000 0000 0000 0000 0000     2a2a 2a2a 2a2a 2a2a 2a2a
           Repeat ...
  254950    0000 e14d 0000 00e1 4d00     ffff e14d 0000 00e1 4d00
  254960    0000 e14d 0000 00e1 4d00     0000 e14d 0000 00e1 4d00
           Repeat ...
  255530    0000 e14d c010  f2f9  00ff     0000 e14d 0000 00e1 4d00
  255540    0100 c980 2cae 59ae ad1e     0000 e14d 0000 00e1 4d00
  255550    b200 0000 e14d 0000 00e1     0000 e14d 0000 00e1 4d00
  255560    4d00 0000 e14d 0000 00e1     0000 e14d 0000 00e1 4d00
           Repeat ...
  262150    4d00 0000 e14d 0000 00e1     0000 e14d 40c0 ecf7 0047
  262160    4d00 0000 e14d 0000 00e1     3e1e b200 0000 e14d 0000
  262170    4d00 0000 e14d 0000 00e1     00e1 4d00 0000 e14d 0000
           Repeat ...
```

Note: To ensure that *linemon* is recording an H.223 level 2 3G-324M call, verify that the received data contains E1 4D 00 00 00.

Recording trunk timeslot 40 in E1 mode

The following example records trunk timeslot 40 using E1 mode.

Command

```
linemon -l 40 -e
```

Received data

```
E1 CAS or PRI mode set (30 timeslots per trunk)
Monitoring timeslot 40 using DSP timeslots 144 (line.out) and 145 (line.in)
Making switch connections 16:40->17:144 (line.out) and 4:10->17:145 (line.in).
Recording of Line Out and Line In Started.
```

Recording timeslot 0 in T1 mode using a specific board and specific monitors

The following example records timeslot 0 on board 1 into *monitor_ts0.out* and *monitor_ts0.in*. It uses T1 mode (the default mode).

Command

```
linemon -b 1 -f monitor_ts0
```

Returned data

```
Monitoring timeslot 0 using DSP timeslots 144 (monitor_ts0.out) and 145 (monitor_ts0.in)
Making switch connections 16:0->17:144 (monitor_ts0.out) and 0:0->17:145
(monitor_ts0.in).
Recording of Line Out and Line In Started.
```

5 h324Extract

Overview of h324Extract

The *h324Extract* utility extracts audio and video data from the recorded TDM bearer channel data. It creates raw audio and video files using data recorded and prepared by the *linemon* utility. If *3gpapp* is accessible from the current directory, you can use *h324Extract* to convert the raw audio and video data to a 3GP file.

h324Extract can decode any 3G-324M call terminated by Video Access for call setup techniques including NSRP, WNSRP, MONA ACP, and MONA MPC.

If a MONA terminal encapsulates initial media frames in MONA signaling preferences messages, those frames are not extracted by *h324Extract*.

Currently, *h324Extract* does not support the creation of 3GP files when the audio codec is G.723, when the video codec is H.264, or both. However, the raw bit streams for audio and video can be extracted for these codec types.

For more information, refer to *Overview of linemon* on page 29 and *Overview of 3gpapp* on page 11.

h324Extract syntax

h324Extract is installed in the *C:\nms\bin\* directory in Windows or the */opt/nms/bin/* directory in UNIX.

h324Extract uses the following syntax:

```
h324Extract [-3gp][-o]<h324 log file> <h223 stream file> [<output file name>]
```

where:

Argument	Description
-3gp	Command to convert the generated audio and video files to 3gp file format. To use this capability, the *3gpapp* utility must be accessible on the same platform.
-o	Indicates the data in the given H.223 stream input file is from the outgoing TDM bearer channel, for example, line.out. If this argument is not specified, it indicates that the data in the given file is from the incoming TDM bearer channel, for example, line.in.
h324 log file	The name of the log file created by the Video Access H.324M Middleware. *h324Extract* uses this log file to extract the audio and video data from the stream.
h223 stream file	The name of the file that contains the recorded TDM bearer channel data. You can use the *linemon* utility to record this data.
output file name	The name of the output file, as follows:

If you...	Output is created in...
Specify a file name	A file with the specified name
Do not specify a file name AND you specify the -3gp command	*out.3gp*
Do not specify a file name AND you do not specify the -3gp command	Two files with the name *out.**xxx***, where ***xxx*** refers to the audio and video codecs in the bearer channel data as follows: • amr – for AMR audio data • 723 – for G.723 audio data • 263 – for H.263 video data • 264 – for H.264 video data • m4v – for MPEG-4 video data

Example: Extracting the media from the incoming stream

The following example assumes the *linemon* utility is used to collect the incoming data from a 3G-324M terminal to *line.in* file and the *h324.log* is the corresponding Video Access H.324M middleware log file.

The following table describes two possible scenarios that assume the session is negotiated with H.263 video codec and AMR audio codec:

This command...	Sends output to...
`h324Extract h324.log line.in`	• *out.amr* (binary file) • *out.263* (binary file)
`h324Extract h324.log line.in test`	• *test.amr* • *test.263*

Example: Creating 3GP files from the incoming stream

The following example assumes the *linemon* utility is used to collect the incoming data from a 3G-324M terminal to *line.in* file and the *h324.log* is the corresponding Video Access H.324M Middleware log file.

The following table describes two possible scenarios that assume the session is negotiated with H.263 video codec and AMR audio codec:

This command...	Sends output to...
`h324Extract -3gp h324.log line.in`	*out.3gp* (assuming the *3gpapp* utility is accessible from the same directory)
`h324Extract -3gp h324.log line.in test.3gp`	*test.3gp* (assuming the *3gpapp* utility is accessible from the same directory)

Example: Creating a 3GP file from the outgoing stream

The following example assumes the *linemon* utility is used to record the outgoing data to a 3G-324M terminal to *line.out* file and the *h324.log* is the corresponding Video Access H.324M Middleware log file.

The following table describes two possible scenarios that assume the session is negotiated with H.263 video codec and AMR audio codec:

This command...	Sends output to...
`h324Extract -3gp h324.log line.out`	*out.3gp* (assuming the *3gpapp* utility is accessible from the same directory)
`h324Extract -3gp -o h324.log line.out test.3gp`	*test.3gp* (assuming the *3gpapp* utility is accessible from the same directory)

Index