Application Note

IP Call Logging Using Dialogic® HMP Software and a Passive IP Tap

Executive Summary

Call logging in a TDM environment usually requires high impedance products that are physically attached to a cable carrying live traffic, but that do not affect the traffic. In an IP environment, additional capabilities can manipulate and replicate the voice signal. However, some of the architectures that are used for IP logging alter the traffic carrying environment and introduce points of failure.

Passive IP monitoring is a method that works much like a high impedance T-1/E-1 tap. The IP tap is physically attached to the Ethernet cable, but it does not alter packet flow. Packets are simply replicated and sent through a separate interface for further processing, not affecting the actual voice transaction between parties.

This application note describes a way of using a tap to build a high density call logging application with Dialogic® HMP Software as the control and recording medium. A working demo application can also be downloaded.

Table of Contents

Introduction 2
    Hardware and Software Components Used 2
    Other Considerations 2
High-Level Architecture 2
IP Tap 4
RTP Stream Aggregation 4
Packet Monitoring and Capture, and Application Control 4
    Packet Capture 4
    Media Streaming to the Logger 5
    The User Interface 5
    PCAP Monitor Application Structure 6
Media Stream Reception and Full Duplex Transaction Recording 6
    RTP Streaming 6
    IP Logger Application Structure 6
Using the Demo Application 7
    Wiring the NetOptics Tap and Ethernet Hub 7
    Installing WinPcap 8
    Configuring the Applications 8
    Building the Applications 9
    Running the Applications 9
Performance 10
Acronyms 11
For More Information 11

Introduction

Hardware and Software Components Used

Capturing IP voice data may seem like an easy task. You may think that since the IP voice data is the payload portion of an RTP packet that you can copy this data, append it to a file and the recording is done.

However, in practice, it is not quite so simple. Each two-party conversation is made up of two half-duplex voice streams that must be properly mixed while keeping level and timing characteristics under control. The RTP streams themselves must be intercepted, either by relaying them through a monitoring and recording system, or by passively monitoring streams without changing their path. Finally, high recording densities must be achieved to maintain a favorable cost-per-port ratio.

This application note describes how to overcome these obstacles by using three hardware and software components:

1. Dialogic® Host Media Processing Software Release 3.0 for Windows®
2. A NetOptics passive monitoring IP tap
3. WinPcap open source IP packet capture software

Dialogic® HMP Software is effective for efficient RTP handling and offers a multitude of RTP coders and recording formats with the ability to transcode between them. Dialogic® transaction recording uses proven DSP algorithms, running on the host processor, to mix the two sides of the half-duplex RTP streams that make up a typical conversation. Dialogic HMP Software has also been designed and tested to achieve an extremely high port density on a PC-based media server.

Passive IP monitoring is necessary to avoid potential disruptions in a call center environment. Using a NetOptics tap is one way to avoid adding further points of failure replicating packets.

Also, a way of delivering, capturing, filtering, and sorting the monitored IP streams before they can be recorded is necessary. This can be accomplished using WinPcap open source IP packet capture software. It provides a high-level interface to a PC's Ethernet interface so that an application programmer does not need to write device driver level code.

Other Considerations

The Windows® operating system was chosen for this application note. However, the code as written can be ported to Linux and the Dialogic® Host Media Processing Software Release 3.1LIN because:

- Dialogic® API calls can be found in Dialogic HMP Software 3.1
- The WinPcap project is an offshoot of the libpcap project, which originated on UNIX/Linux
- UDP socket commands used for inter-process communications are also available on Linux

If a passive monitoring tap is not available, this demo can still be run using just an Ethernet hub. The tap simply allows a way of accessing an IP stream without introducing an additional point of failure. A real implementation of this demo and all functional and load testing should be done with a tap, but it is not necessary for a proof of concept.

While this demo does not handle Secure RTP (SRTP), it could be modified to do so. Dialogic® Internet Protocol Media Library (IPML) can be configured to decrypt packets when they are received, allowing unencrypted voice to be recorded. It would be necessary to capture and set SRTP parameters, and especially, the encryption key for each stream.

High-Level Architecture

A large number of half-duplex RTP packets are used to carry multiple IP conversations. To go from these packets to the desired collection of full-duplex recordings in the desired format requires many components and steps. To help explain this, the components that make up the IP Call Logging System described in this application note can be divided into four basic parts. They are pictured in Figure 1. The circled numbers refer to the paragraphs that follow.

1. **IP Tap** — Packets carrying voice traffic are physically replicated by an IP tap before they are fed to a system where they can be sorted, arranged, and stored.
2. **RTP Stream Aggregation** — Output from the IP tap consists of two series of RTP streams, one from each twisted pair of wires, and each going in a different direction. These streams must be aggregated to provide a single interface to the packet capture software. Aggregation is done using an Ethernet hub in the diagram.

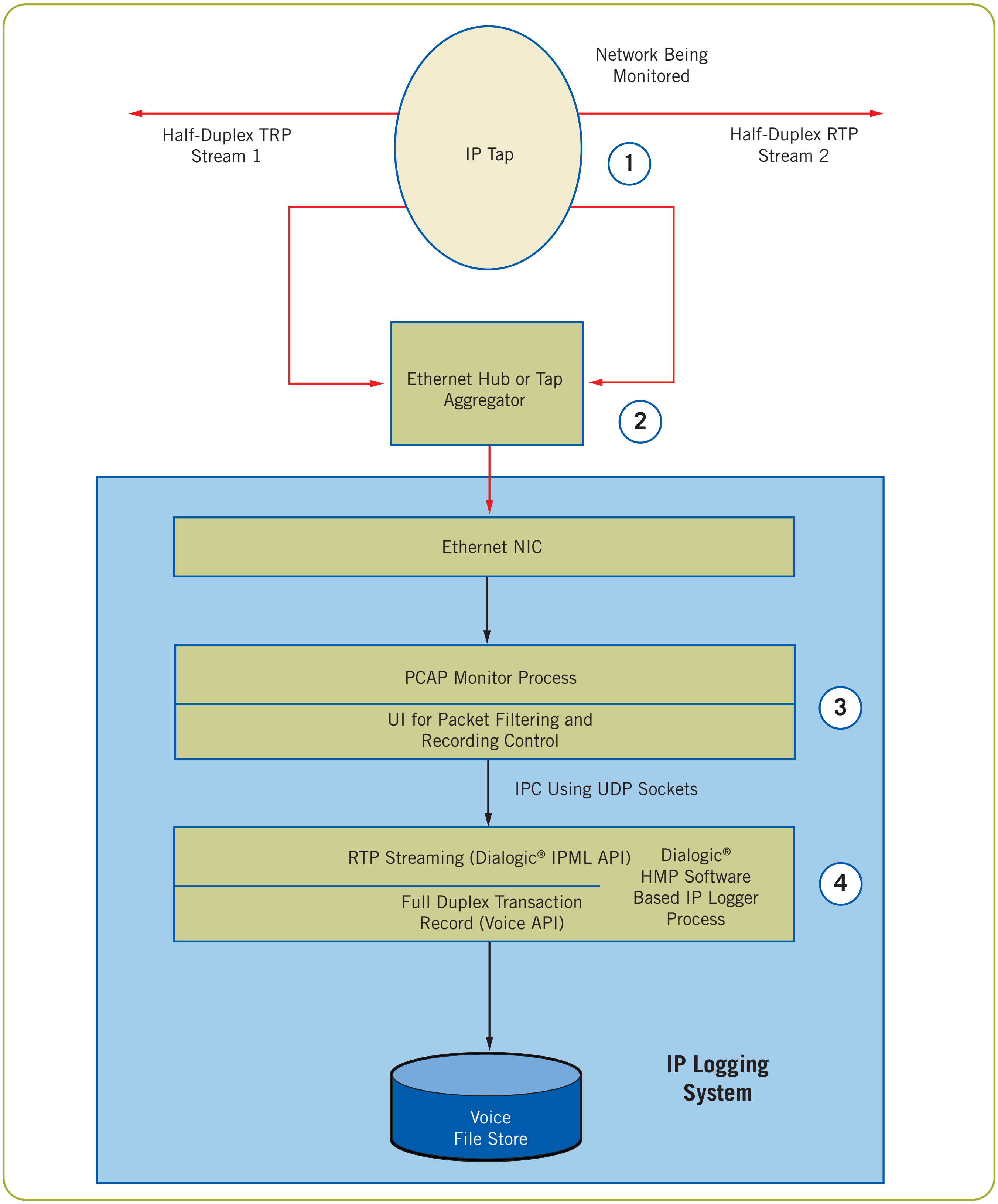
Network Being Monitored
Half-Duplex TRP Stream 1
IP Tap
Half-Duplex RTP Stream 2
1
Ethernet Hub or Tap Aggregator
2
Ethernet NIC
PCAP Monitor Process
UI for Packet Filtering and Recording Control
3
IPC Using UDP Sockets
RTP Streaming (Dialogic® IPML API)
Full Duplex Transaction Record (Voice API)
Dialogic® HMP Software Based IP Logger Process
4
Voice File Store
IP Logging System

Figure 1. High-Level Architectural View

3. **Packet Monitoring and Capture, and Application Control (PCAP Monitor process)** — A low-level software interface to the PC's Ethernet card is needed to efficiently access the packets. To control this, a user interface specifies which packet streams should be recorded among those available, and starts and stops monitoring and recording.
4. **Media Stream Reception and Full Duplex Transaction Recording (IP Monitor process)** — A media logging component is used for transporting, combining, and recording the conversations.

In the following sections, each part of the system is discussed.

IP Tap

NetOptics makes a full line of passive monitoring IP taps for both copper and optical, from 10Base-T to 10 Gigabit, with one to twenty links per chassis. A single link portable tap, the "10/100 Base-T Teeny Tap", was used in this application note's demo. All Ethernet traffic in the demo traveled at a 100Base-T rate over copper Cat 5 cable.

The idea of "passive monitoring" is important in any call center or other complex IP environment. The tap does not introduce delay into the packet stream, and in the event of a power failure or other problem with the tap, the original stream remains untouched. Only the monitoring will stop functioning.

RTP Stream Aggregation

The NetOptics tap has two separate monitoring ports, one for each twisted pair in the monitored Cat 5 cable. So, the packets going in one direction appear on one port and packets going in the other direction on the other port. These two streams must be combined to be made available to the packet monitoring software, which works from a single Ethernet NIC.

For this application note, a common Ethernet hub was used for stream aggregation. A hub reflects the packets seen on any one port on all other ports, and lets the device attached to a port decide whether to accept or ignore the packet. This can lead to excessive traffic in most situations, but it is what is needed for IP call logging where a single port wants access to packets going to and from many different addresses.

If a system has two Ethernet cards, it may be possible to use the two interfaces in a teamed (Windows®) or bonded (Linux) mode so that they appear as a single interface to the packet capture utility.

A third possibility is to use a tap with a built-in aggregator. This provides all monitored packets on a single Ethernet interface.

Packet Monitoring and Capture, and Application Control

The PCAP Monitor process is an independent, multi-threaded part of the application that receives the aggregated packets from the tap on the Ethernet connection being monitored. It also contains a console-based user interface. The process performs the following functions:

- Sets up the low-level interface to the Ethernet NIC card
- Sets up RTP streams to be captured
- Starts/stops capture
- Relays the desired streams to the logging process
- Sends record start/stop commands to the logging process
- Tells the logging process when to terminate

Packet Capture

Packet capture is done using the WinPcap open source library, which is described on its website (see the *For More Information* section) as follows:

"WinPcap is the industry-standard tool for link-layer network access in Windows environments: it allows applications to capture and transmit network packets bypassing the protocol stack, and has additional useful features, including kernel-level packet filtering, a network statistics engine and support for remote packet capture."

As such, the library forms the basis for several Ethernet monitoring tools, such as WireShark and TCPDump/WinDump. It also makes a base for the IP Logger process.

The PCAP Monitor process uses the WinPcap library as follows:

1. Opens the WinPcap library and queries the system for available Ethernet interfaces.
2. Opens the selected interface.

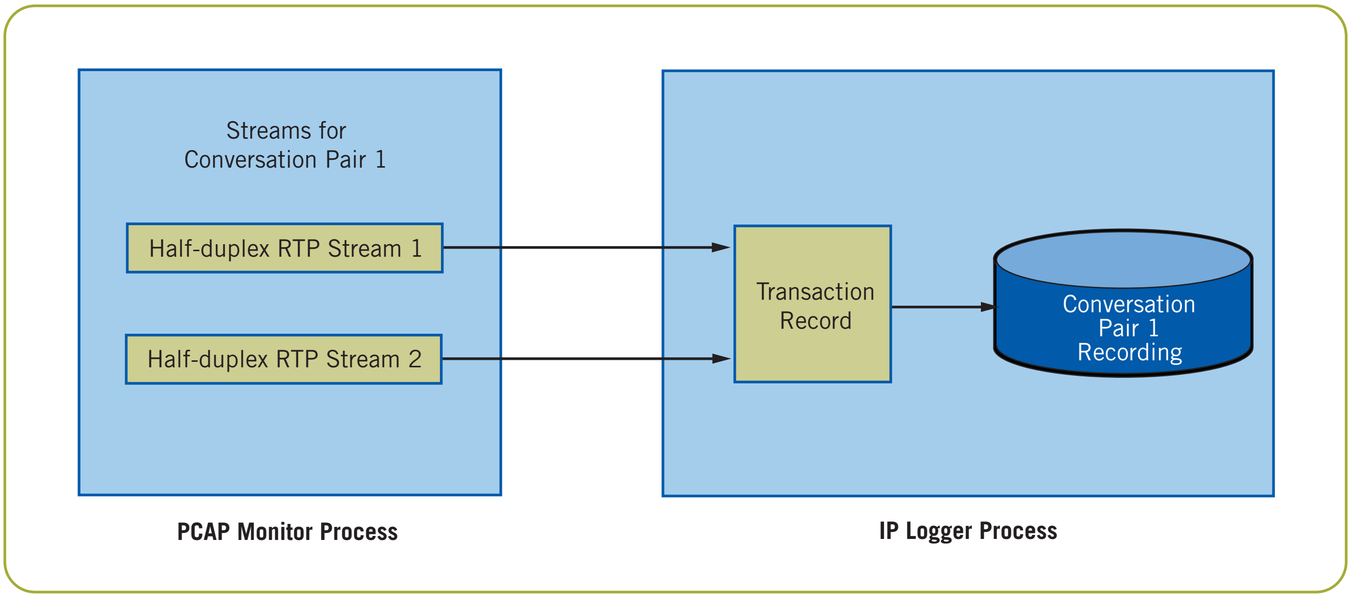

Figure 2. RTP Streams and Transaction Record

3. Clears, sets, compiles, and activates a filter to return only UDP packets to the application.
4. Starts packet monitoring in a callback mode, where each packet detected on the Ethernet interface arrives in a user-defined callback routine.

Packet Filtering

WinPcap has a packet filtering feature that allows the programmer to give the specifics of the type of packets to be captured. An RTP stream can be uniquely identified by:

- Source IP address
- Source UDP port
- Destination IP address
- Destination UDP port

Although a single RTP stream does not require much information, IP logging requires two streams per conversation, and a high density system may monitor several hundred conversations. This would lead to a very complicated filter specification. So, the WinPcap filter is set to distinguish only TCP from UDP packets (RTP is always UDP).

Identifying and grouping streams for logging is done instead by the Pcap Monitor process. The unique identifier for each stream — source/destination IP address and UDP port –- is combined into a 12-byte key. This key is used as a lookup for a Standard Template Library (STL) map container. The map associates the key with a pair of open connections to the logging process. So, for each UDP packet that arrives on the Pcap interface, an efficient lookup is done to see if that packet should be passed on to the logger. If not, it is discarded.

Media Streaming to the Logger

The HMP-based IP Logger process is set to receive some number of RTP streams on well-defined UDP ports. Since the ports are predictable given the number of channels in use, just send the packets making up an RTP stream to the desired address/port using UDP socket interface calls. There the stream is received and can be recorded.

Remember that a two-way IP conversation consists of a pair of half-duplex RTP streams, one in each direction. The pair of captured streams is associated with one another in the PCAP Monitor process, but sent as separate streams. They are only combined into a single recording using transaction record in the IP Logger process. This is shown in Figure 2.

Thus, the PCAP Monitor process works in terms of “Pair IDs”. These IDs are a single number that identifies a pair of RTP streams that make up a two-way conversation.

The User Interface

The PCAP Monitor process also contains a text-based user interface. It is comprised of a single level menu that will query for additional data for a given choice.

```
Main Menu
=========
1) Interactively create a new recording session
2) Create recording sessions from config file
3) Display recording sessions
4) Delete an existing recording session
5) Start PCAP Monitoring
6) Stop PCAP Monitoring
7) Quit

Enter choice:
```

Menu choices and operation will be explained in the *Running the Application* section.

PCAP Monitor Application Structure

The structure of the PCAP Monitor application is relatively straightforward. A single thread provides the interactive menus and processes commands entered. When packet monitoring is started, a second thread is spawned to handle the WinPcap API calls. The final call starts the monitoring and a callback routine is invoked for each packet detected. There the log/nolog decision is made. Packets to be logged are sent out the correct
UDP socket port to the logging process. Other packets are ignored.

Media Stream Reception and Full Duplex Transaction Recording

The IP Logger process contains Dialogic® RTP streaming and voice recording. Also, a UDP socket connection receives and interprets commands to start and stop recording, and to end the process.

RTP Streaming

Dialogic RTP streaming is achieved through the Dialogic® IP Media Library (IPML) API. This API does not do SIP signaling. Rather, it specifies UDP ports for socket connections, voice coders, and other RTP-related information for each of the two half-duplex streams. When the ipm_StartMedia() command is issued, RTP streaming begins in a receive-only direction, since there is no need to send any voice data in the other direction. Voice from the remote end is recorded.

IP Logger Application Structure

The IP Logger is a single-threaded, multi-channel, asynchronous application. After all desired IPML and voice devices are opened, the application goes into an event processing loop.

Events come from one of two sources. The first is a Dialogic device event, such as media stream started or recording done. The second is a command from the PCAP Monitor's user interface.

Here are the three simple commands that are interpreted by the IP Logger:

1. "B IPMLPortNumber FileName" – Begin receiving RTP and recording on IPML Port X. The first command on this port starts receiving the first half-duplex stream. The second command on this port starts receiving the second half duplex stream and starts full duplex transaction record on the pair of streams.
2. The recording is given the file name in the command "S IPMLPortNumber" – Stop receiving RTP on the pair of streams and terminate the recording; a voice file results.
3. "E" – End all ongoing streams and recordings, close all devices, and terminate the process.

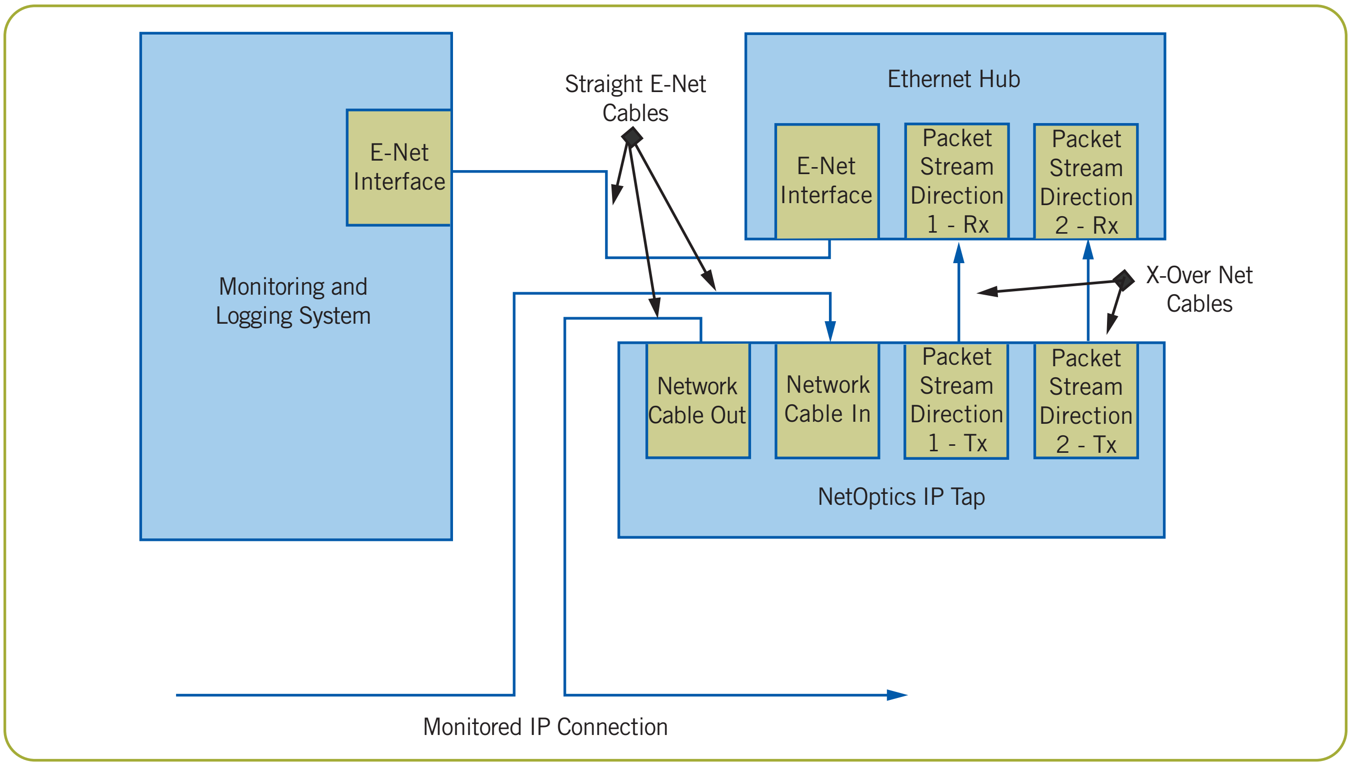

Figure 3. Wiring the NetOptics IP Tap

There are multiple IP Logger objects, one for each pair of IPML connections with a single voice device for doing transaction recording. The object that handles the incoming event is looked up either by device handle for a Dialogic event or by IPML port number in the case of a command from the PCAP monitor.

Each IP Logger object is built around a simple object-oriented state machine. The three states are opening, idle, and recording. A logger is *opening* when the two IPML connections and one voice device are being opened and connected on the soft CTBus; a logger is idle when it is ready to record; and it is recording when RTP streaming and transaction recording is underway.

The application cleanly terminates either on command from the PCAP monitor or when it receives a Ctrl-C interrupt from the keyboard.

The RTP coder and recording format used for this demo are:

- G.711 — The direct packetization of a 64 kBps μ law PCM. It is expected that in an internal call center environment, G.711 would be the coder used.
- Dialogic ADPCM — A likely choice of recording file formats although, any RTP coder used by the Dialogic IPML API and any recording file format supported by the Dialogic® Voice API may be used. Disk space is likely to be less of a consideration than the CPU cycles needed to do audio compression.

Using the Demo Application

The C++ code for the PCAP Monitor and IP Logger processes may be downloaded along with this application note (see the *For More Information* section). The remainder of this section describes how to configure, build, and run the IP Logger.

Wiring the NetOptics Tap and Ethernet Hub

Figure 3 shows the correct wiring to deliver the packets on the tapped line to the monitoring system. Note in particular that the two cables connecting the NetOptics IP tap and the Ethernet hub are crossover cables, where the transmit wire pair is connected to the receive pins on one side, and the receive wire pair is connected to the transmit pins on the other.

Installing WinPcap

WinPcap 4.0.1 must be installed before the application is built. Run the installation wizard for a straightforward installation.

Configuring the Applications

Before building the applications, edit the include files, configheader.h. Configure site-specific addresses and parameters in this file. Only change ports if the specified port is already in use.

Since the PCAP Monitor and IP Logger are two independent processes connected by UDP sockets, they may either be run on the same or separate systems. Splitting across two systems would allow for higher densities, but a dedicated LAN would likely be necessary to carry RTP traffic between them. In either case, the ports and addresses should be set in the configheader.h file.

Stream configuration is done in the PCAP Monitor process. Each RTP stream to be monitored can be uniquely described by its source and destination IP addresses and UDP ports. As previously mentioned, this data is used for deciding which packets to keep and where to send them.

There are two ways to specify how these streams should be recorded. The first is to manually enter the IP addresses and ports needed to describe the two streams that make up the conversation. This is described in the *Running the Applications* section.

A second and generally more convenient way is to use the configuration file, SessionDefs.cfg. In this file, each stream's IP address and port information is on a separate line. In addition, each pair of streams is given the same "Pair ID" so that they will be mixed together in the transaction recording.

Here is a file configured to do four separate recordings:

```
######################################################################
#
# SessionDefs.cfg
#
# A convenient way to specify RTP stream pairs to be captured
# using PCAPMonitor.
#
# Each pair is a one-way RTP stream representing 1/2 of the full-duplex
# conversation being recorded
#
# Entries must be in this format:
# Pair ID      Src IP      Src UDP     Dst IP      Dst UDP
#        Addr        Port        Addr        Port
#
# Can be whitespace or comma delimited
######################################################################
1  192.168.1.105     49152       192.168.1.118     49152
1  192.168.1.118     49152       192.168.1.105     49152
2  192.168.1.105     49154       192.168.1.118     49154
2  192.168.1.118     49154       192.168.1.105     49154
3  192.168.1.105     49156       192.168.1.118     49156
3  192.168.1.118     49156       192.168.1.105     49156
4  192.168.1.105     49158       192.168.1.118     49158
4  192.168.1.118     49158       192.168.1.105     49158
```

Note: It will be necessary to find the ports used by each IP endpoint before a conversation can be recorded. Many SIP soft phones are able to log SIP messages, and the ports can be seen in the SDP portion of INVITE and OK messages. Using an IP "sniffer," such as Wireshark or TCPDump, is another possibility. Phones also tend to have a pattern in selecting their RTP ports. You may be able to see it if you are familiar with the phone.

In a production application, a less error-prone and more user friendly way to specify which calls to record would be used. However, the outcome would still need to produce the same addresses and ports for the two streams.

Building the Applications

The two Windows® applications were developed using Microsoft® Visual Studio® 2005. Both demo projects, IPLogger and PCAPMonitor, should be built. The Debug configuration is active for each. PCAPMonitor will need to reference the `PCAP\WpdPack\Include` directory from the the PCAP library.

It is also possible to use Microsoft's free introductory development environment, Visual Studio Express Edition, with Dialogic HMP Software applications. The Visual Studio 2005 project files will work with the Express Edition.

Running the Applications

Ensure you have sufficient Dialogic HMP Software licenses. One voice resource license and two basic RTP licenses are needed for each channel.

The IP Logger process should be started first. It takes a single argument — the maximum number of channels/ conversations to be recorded:

```
        C:\iplogger> iplogger 60
```

The PCAP Monitor process is started without arguments.

Two things must be done before recordings can be activated. First, WinPcap packet monitoring must be started. To do this, select the following menu choice:

```
        5) Start PCAP Monitoring
```

This first checks the system for any Ethernet interfaces, and allows the operator to choose the interface to be monitored.

```
Ethernet adaptors found:
        1. \Device\NPF_GenericDialupAdapterAdaptor description: Adapter for
           generic dialup and VPN capture
        2. \Device\NPF_{2FAA7125-FB06-4F0D-A136-9CB4F833D325} Adaptor
           description: Intel(R) PRO/1000 MT Dual Port Network Connection
        3. \Device\NPF_{81F5AF1C-A42B-4097-87EF-46C40A22170C} Adaptor
           description: Intel(R) PRO/1000 MT Dual Port Network Connection
        Enter the interface number (1-3): 3
```

Once the interface is selected, packets are being captured, but not yet recorded. To do this, at least one IP address/port pair must be entered. This can be done manually by selecting the following from the main menu:

```
        1) Interactively create a new recording session
```

Source and destination IP addresses and UDP ports for both half duplex streams must be entered. If many ports are being recorded, it will be more convenient to list the address/port data in the SessionDefs.cfg file, as described in the *Configuring the Applications* section. The file is activated by selecting the following from the main menu:

```
        2) Create recording sessions from config file
```

Configuration from a file may only be done once.

When session entry is finished, the streams being recorded can be viewed by selecting the following from the menu:

```
    3) Display recording sessions
```

Pairs of monitored streams will be displayed in this fashion:

```
    Ongoing Recording Sessions
    ===========================
    Pair ID Src IP Addr/Port                Dst IP Addr/Port
    ________________________________________-
          1        192.168.1.117/49200      192.168.1.109/49200
          1        192.168.1.109/49200      192.168.1.117/49200

          2        192.168.1.117/49202      192.168.1.109/49202
          2        192.168.1.109/49202      192.168.1.117/49202
```

Recording continues until it is stopped. Sessions can be stopped individually with this menu choice:

```
    4) Delete an existing recording session
```

This selection displays a listing of all active sessions. The paired sessions are displayed and can be selected to stop the session:

```
    Ongoing Recording Sessions
    ===========================
    Pair ID Src IP Addr/Port          Dst IP Addr/Port
    _______________________-
    1        192.168.1.117/49200      192.168.1.109/49200
    1        192.168.1.109/49200      192.168.1.117/49200

    2        192.168.1.117/49202      192.168.1.109/49202
    2        192.168.1.109/49202      192.168.1.117/49202
    Session pair ID to delete? 2
```

Finally, all packet monitoring, recording, and both processes can be shut down with a single command:

```
    7) Quit
```

A Ctrl-C will shut down either process individually.

The resulting recording files are named *Recording_X.vox,* where X is the Pair ID. They may be replayed with an audio utility such as CoolEdit or Audacity.

Performance

A series of test runs were done using various recording densities. The following system was used to monitor and record:

- Intel rack-mount server
- Dual Xeon 2.4 GHz Hyperthreaded CPUs
- 1 GB of RAM
- 100 Base-T Netgear DS108 Hub
- 1000 Base-T Ethernet interface, running at 100 Base-T to match Hub

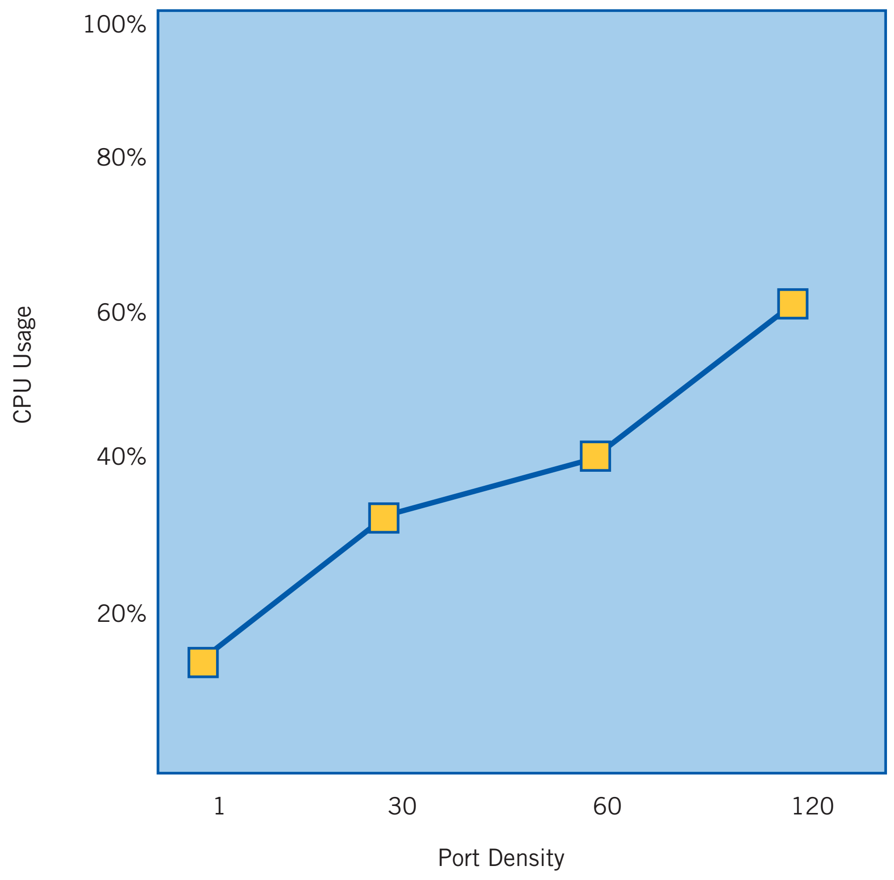

Figure 4. Port Density versus CPU Usage

To represent a realistic traffic load from which calls are recorded, 240 ongoing RTP stream pairs were generated between two other systems using Dialogic HMP Software. The NetOptics tap was placed in-line between the two traffic generation systems. The number of simultaneous recordings achieved is shown in Figure 4, along with the CPU utilization for each density.

In all cases, the PCAP Monitor process had the same number of packets to sort through – 240 half duplex streams. Thus, a minimum of 16% CPU was needed to record even a single conversation. CPU usage when adding additional recordings was fairly linear, up to 60% for 120 ports.

Higher densities are possible. The main limitation would be the number of user sessions — an RTP and voice resource — supported by Dialogic HMP Software. Recall that two IPML streams are used by the IP Logger process. This would bring a system loaded to 250 ports up to the current maximum supported number of 500 user sessions.

Acronyms

HMP	Host Media Processing
IP	Internet Protocol
IPML	Internet Protocol Media Library
LAN	Local Area Network
NIC	Network Interface Card
PCAP	Packet Capture
RTP	Real-time Transport Protocol
SIP	Session Initiation protocol
SRTP	Secure Real Time protocol
TDM	Time Division Multiplexed
UDP	User Datagram Protocol

For More Information

A Zip file containing the demo projects can be downloaded at http://www.dialogic.com/goto/?10728

WinPcap: The Windows Packet Capture Library — http://www.winpcap.org/

PCAP Library for Linux — http://www.tcpdump.org/

NetOptics Passive IP Monitoring — http://netoptics.com

Dialogic®

To learn more, visit our site on the World Wide Web at **http://www.dialogic.com**.

Dialogic Corporation
9800 Cavendish Blvd., 5th floor
Montreal, Quebec
CANADA H4M 2V9