thinklogical

Router Interfaces

Thinklogical, LLC ®
100 Washington Street
Milford, Connecticut 06460 U.S.A.
Telephone : 1-203-647-8700
Fax : 1-203-783-0049
www.thinklogical.com

Value Your Content
thinklogical®
Trust Our Proven Ingenuity

Revision: H

Table of Contents

Index of Tables

Preface

This document describes the serial and network interfaces for Thinklogical's® VX/MX/HDX family of routers. There is more extensive documentation available concerning the ASCII interface API, SNMP MIB definitions, and additional information at the Thinklogical® ftp site: ftp://ftp.thinklogical.com/VXRouter/Documentation/. Other manuals and QuickStart Guides may be downloaded from our web site: http:/www.thinklogical.com/resources.

These manuals are available for downloading:

- **Router-ASCII-API.pdf**
- **Configuring-the-ASCII-Interface.pdf**
- **Router_SNMP_Traps.pdf**
- **IP_Address_Change_Guide.pdf**
- **Manual_How To Change A Router's IP Address.pdf**

Router SNMP MIB definition files that are available for downloading:

- **LSI-ROOT.mib**
- **LSI-ROUTER-API-INTERFACE.mib**
- **LSI-SFP.mib**
- **LSI-VXROUTER.mib**

Byte Order

The 16-bit values documented here store the most significant byte first. This is called Big Endian format. For example, a hexadecimal value of 1235H is stored with 12H in byte 0 and 23H in byte 1.

If your receiving system is not a Big Endian CPU (Intel processors are not), then you will have to swap the byte order before you can use the value. If you don't swap the bytes, then a value of 1 from the router will be interpreted as 256, 2 as 512, and 640 as 32770.

Serial/RS-232

The router CPU card has two RS-232 serial ports. One is for the Linux command line interface (labeled CONSOLE), and the other is for the router's ASCII command interface (labeled RS232). If a back-up CPU is installed, then its Linux console port is active. However, only the 'RS232' port on the active CPU will function. The active CPU is the one with the FAULT LED turned off.

Linux Command Console (Console)

- Baud Rate: 115200
- Data Bits: 8
- Parity: none
- Stop Bits: 1
- Flow Control: none
- DB9 DCE
  - A straight (NOT a null-modem) cable is needed to connect to a PC.

ASCII API (RS232)

- Baud Rate: 9600
- Data Bits: 8
- Parity: none
- Stop Bits: 1
- Flow Control: XON/XOFF (Software)
- DB9 DCE
  - A straight (NOT a null-modem) cable is needed to connect to a PC.

Network

The router will use up to three IP addresses. The primary CPU card will use address X and address X+100. The back-up CPU will use address X+1 and it will take over address X in the event of a failure in the primary CPU.

The factory default value for 'X' is 192.168.13.15. Steps for changing the IP address are described in the router manual: **VX40_160_320_320V_Manual_Rev_*.pdf**.

The router has several network ports reserved for internal use. Port 17567 accepts commands to control the switch. The command API is described in the manual: **VxRouter-ASCII-API.pdf**.

There are several other ports that can be read to access system configuration and operating data. The actual data format varies by model and is described in the next sections. There are five blocks of data available for reading.

1. Switch connection status (broadcast over port 17564 every few seconds)
    - The primary CPU IP address
    - Upper or Lower chassis flag
    - First port number in this block
    - Last port number in this block
    - 'N' port number values
2. Port settings (UDP port 17565, router localhost only)
    - Port output level value
    - Port input enable status
    - Input port number (big-endian format)
3. Hardware sensor status (TCP port 17566)
    - This is the text output from the lm_sensors 'sensors' command and is reserved for future use.
4. Alarm, Inventory and Power Supply Status (TCP port 17600)
    - State of the alarm contacts
    - Card Inventory
    - Reset Bits
    - Power Supply Status
5. I/O card data (TCP ports 17601 through 17640, 1 port per I/O card)
    - SFP TxDisable & TxFault bits
    - SFP installed flag
    - I/O card type
    - SFP serial ID data
    - SFP Diagnostic data

Switch connection status is broadcast over the network every 4 seconds. The information in the data block is not sufficient to identify the model of the router. You should use the IP address in the data to identify which router sent the message.

In the future, newer models may have more I/O ports than can be contained in one data block. In that event, the blocks will be split up into several pieces, and the first and last port numbers will be used to reassemble that data.

The data in the status block is stored in big-endian format. This means the MSB of the data is stored first, and then the LSB. This is **NOT** the standard for x86 based data storage.

The other ports must be opened as a TCP connection and then have the data read. After reading the data, you MUST then close the connection.

There are several useful network applications running that may aid in monitoring the hardware. Three of these are: SNMP, NTP and syslog. These are described in subsequent sections.

Multicast vs. Broadcast

Version 4.5 of the API and version 2.05 of the LOS manager now offer the option of using multicast in place of broadcast for sending status messages. Multicast will use the same ports as broadcast; 17564 for the API and 17560 for the LOS manager. The default multicast IP address is 239.255.13.9.

References in this document to broadcast will be valid for multicast.

17564 – Switch Connection Status Broadcast

- 32-bits, router's primary IP address
- 16-bits, 0 for the Upper (only) router chassis
- 16-bits, first port number in the list (list starts from 1, not 0)
- 16-bits, last port number in the list
- N * 16-bits, the input port that is connected to output port X, zero means the port is NOT connected

Example: `c0 a8 0d 19  00 00  00 01  00 50  00 11  00 12…`

IP address = 192.168.13.25
Chassis = 0
First Port number = 1
Last Port number = 80 (`0x0050`)
Output 1 is connected to input 17 (`0x0011`)
Output 2 is connected to input 18 (`0x0012`)

17565 – Port Settings (localhost only)

80 sets of the following data:

- (8 bits) output level 0 is off
- (8 bits) input level
- (8 bits) connected to input port number, starting from 0

A 'C' data structure for this would be:

```
struct s_block {
        unsigned char  output_level;
        unsigned char  input_level;
        unsigned char  connectedTo;
} data_block [80];
```

17566 – Hardware Status

Reserved for future use.

17600 – System Data

System data consists of 8 bytes (12 for the HX80). The first byte has the current alarm contact settings. Each bit corresponds to a contact. Bit 0 (lsb) is the bit for the Power Supply 1 alarm, and bit 7 (msb) is the bit for the 'any' alarm. The next two bytes are for the installed card inventory. A '0' bit means the card is installed and a '1' bit means the card is removed. The other bytes are for internal use.

	Alarms	Inventory	
Bit	Byte 0	Byte 1	Byte 2
0	Alarm 1	Card 1	Card 9
1	Alarm 2	Card 2	Card 10
2	Alarm 3	Card 3	Card 11
3	Alarm 4	Card 4	Card 12
4	Alarm 5	Card 5	Card 13
5	Alarm 6	Card 6	Card 14
6	Alarm 7	Card 7	Card 15
7	Alarm 8	Card 8	Card 16

Table 1: VX40/VX80/HX80 System Data

The HX80 has 4 additional bytes at the end for sync card status.

17601:17616 – Card Data

Each I/O card in the router has an associated TCP port. Card 1 uses port 17601, card 2 uses 17602, and so on. The data is formatted as follows:

- 6 bytes, internal use
- 1 byte, card type
  - `0xff` missing
  - `0x00` upstream
  - `0x01` downstream (only the VX40 has downstream cards)
- SFP data, 256 bytes per SFP, 5 SFPs per card
  - 128 bytes, SFP ID data
  - 128 bytes, SFP diagnostic data

VX160

17564 – Switch Connection Status Broadcast

- 32-bits, VxRouter's primary IP address
- 16-bits, 0 for the Upper (or only) VxRouter chassis
  these are the Green to Blue connections (forward direction)
  1 for the Lower VxRouter chassis
  these are the Blue to Green connections (back direction)
- 16-bits, first port number in the list (list starts from 1, not 0)
- 16-bits, last port number in the list
- N * 16-bits, the input port that is connected to output port X, zero means the port is NOT connected

Example: `c0 a8 0d 19  00 00  00 01  00 a0  00 a0  00 9f…`

IP address = 192.168.13.25
Chassis = 0
First Port number = 1
Last Port number = 160 (`0x00a0`)
Output 1 is connected to input 160 (`0x00a0`)
Output 2 is connected to input 159 (`0x009f`)

The Vx160 consists of two chassis, so there will be two blocks transmitted; one for the upper chassis and one for the lower.

17565 – Port Settings (localhost only)

160 sets of the following data:

- (8 bits)*2 output level		0 is off
- (8 bits)*2 input level
- (8 bits)*2 connected to input port number, starting from 0

The Vx160 consists of two chassis, so there are two entries for each piece of data. The first is the value for the upper chassis and the second is for the lower chassis.

A 'C' data structure for this would be:

```
struct s_block {
        unsigned char output_level[2];
        unsigned char input_level[2];
        unsigned char connectedTo[2];
} data_block [160];
```

17566 – Hardware Status

Reserved for future use.

17600 – System Data

System data consists of 8 bytes. The first byte has the current alarm contact settings. Each bit corresponds to a contact. Bit 0 (lsb) is the bit for the Power Supply 1 alarm, and bit 7 (msb) is the bit for the 'any' alarm. The next two bytes are for the installed card inventory. A '0' bit means the card is installed and a '1' bit means the card is removed. The other bytes are for internal use.

	Alarms	Inventory	
Bit	Byte 0	Byte 1	Byte 2
0	Alarm 1	Card 1	Card 9
1	Alarm 2	Card 2	Card 10
2	Alarm 3	Card 3	Card 11
3	Alarm 4	Card 4	Card 12
4	Alarm 5	Card 5	Card 13
5	Alarm 6	Card 6	Card 14
6	Alarm 7	Card 7	Card 15
7	Alarm 8	Card 8	Card 16

Table 2: VX160 System Data

17601:17616 – Card Data

Each I/O card in the VxRouter has an associated TCP port. Card 1 uses port 17601, card 2 uses 17602, and so on. The data is formatted as follows:

- 15 bytes, internal use
- 1 byte, card type
  - `0xff` missing
  - `0x00` upstream
  - `0x01` downstream
- SFP data, 256 bytes per SFP, 20 SFPs per card
  - 128 bytes, SFP ID data
  - 128 bytes, SFP diagnostic data

17564 – Switch Connection Status Broadcast

- 32-bits, VxRouter's primary IP address
- 16-bits, 0 for the Upper (or only) VxRouter chassis
  1 for the Lower VxRouter chassis
- 16-bits, first port number in the list (list starts from 1, not 0)
- 16-bits, last port number in the list
- N * 16-bits, the input port that is connected to output port X, zero means the port is NOT connected

Example: `c0 a8 0d 19  00 01  00 01  01 40  00 a0  01 40…`
IP address = 192.168.13.25
Chassis = 1
First Port number = 1
Last Port number = 320 (`0x0140`)
Output 1 is connected to input 160 (`0x00a0`)
Output 2 is connected to input 320 (`0x0140`)

The Vx320 consists of two chassis, so there will be two blocks transmitted; one for the upper chassis and one for the lower.

17565 – Port Settings (localhost only)

320 sets of the following data:

- (8 bits)*2   output level   0 is off
- (8 bits)*2   input level
- (16 bits)*2 connected to input port number, starting from 0, big-endian format

The Vx320 consists of two chassis, so there are two entries for each piece of data. The first is the value for the upper chassis and the second is for the lower chassis.

A 'C' data structure for this would be:

```
struct s_block {
        unsigned char output_level[2];
        unsigned char input_level[2];
        unsigned short       connectedTo[2];       /* assumes short is 16 bits */
} data_block [320];
```

17566 – Hardware Status

Reserved for future use.

17600 – System Data

System data consists of 18 bytes. The first two bytes have the current alarm contact settings (big-endian). Each bit corresponds to a contact. Bit 0 of byte 1 is the bit for the Power Supply 1 alarm, and bit 1 of byte 0 is the bit for the 'any' alarm. The next five bytes are for the installed card inventory. A '0' bit means the card is installed and a '1' bit means the card is removed. The other bytes are for internal use.

	Alarms		Inventory				
Bit	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
0	Alarm 9	Alarm 1	Upper Card 1	Upper Card 9	Upper Card 17	Lower Card 5	Lower Card 13
1	Alarm 10	Alarm 2	Upper Card 2	Upper Card 10	Upper Card 18	Lower Card 6	Lower Card 14
2		Alarm 3	Upper Card 3	Upper Card 11	Upper Card 19	Lower Card 7	Lower Card 15
3		Alarm 4	Upper Card 4	Upper Card 12	Upper Card 20	Lower Card 8	Lower Card 16
4		Alarm 5	Upper Card 5	Upper Card 13	Lower Card 1	Lower Card 9	Lower Card 17
5		Alarm 6	Upper Card 6	Upper Card 14	Lower Card 2	Lower Card 10	Lower Card 18
6		Alarm 7	Upper Card 7	Upper Card 15	Lower Card 3	Lower Card 11	Lower Card 19
7		Alarm 8	Upper Card 8	Upper Card 16	Lower Card 4	Lower Card 12	Lower Card 20

Table 3: Vx320 System Data

17601:17640 – Card Data

Each I/O card in the VxRouter has an associated TCP port. Card 1 uses port 17601, card 2 uses 17602, and so on. The data is formatted as follows:

- 12 bytes, internal use
- 1 byte, card type
  - `0xff` missing
  - `0x02` installed
- SFP data, 256 bytes per SFP, 16 SFPs per card
  - 128 bytes, SFP ID data
  - 128 bytes, SFP diagnostic data

VX320Video

17564 – Switch Connection Status Broadcast

- 32-bits, VxRouter's primary IP address
- 16-bits, 0 for the Upper (only) VxRouter chassis
- 16-bits, first port number in the list (list starts from 1, not 0)
- 16-bits, last port number in the list
- N * 16-bits, the input port that is connected to output port X, zero means the port is NOT connected

Example: `c0 a8 0d 19  00 00  00 01  01 40  00 11  00 12…`

IP address = 192.168.13.25
Chassis = 0
First Port number = 1
Last Port number = 320 (`0x0140`)
Output 1 is connected to input 17 (`0x0011`)
Output 2 is connected to input 18 (`0x0012`)

17565 – Port Settings (localhost only)

320 sets of the following data:

- (8 bits) output level   0 is off
- (8 bits) input level
- (16 bits) connected to input port number, starting from 0, big-endian format

A 'C' data structure for this would be:

```
struct s_block {
        unsigned char  output_level;
        unsigned char  input_level;
        unsigned short        connectedTo;  /* assumes a short is 16 bits */
} data_block [320];
```

17566 – Hardware Status

Reserved for future use.

17600 – System Data

System data consists of 18 bytes. The first two bytes have the current alarm contact settings (big-endian). Each bit corresponds to a contact. Bit 0 (lsb) of byte 1 is the bit for the Power Supply 1 alarm, and bit 7 (msb) is the bit for the 'any' alarm. The next three bytes are for the installed card inventory. A '0' bit means the card is installed and a '1' bit means the card is removed. The other bytes are for internal use.

	Alarms		Inventory		
Bit	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
0		Alarm 1	Card 1	Card 9	Card 17
1		Alarm 2	Card 2	Card 10	Card 18
2		Alarm 3	Card 3	Card 11	Card 19
3		Alarm 4	Card 4	Card 12	Card 20
4		Alarm 5	Card 5	Card 13	
5		Alarm 6	Card 6	Card 14	
6		Alarm 7	Card 7	Card 15	
7		Alarm 8	Card 8	Card 16	

Table 4: Vx320Video System Data

17601:17620 – Card Data

Each I/O card in the VxRouter has an associated TCP port. Card 1 uses port 17601, card 2 uses 17602, and so on. The data is formatted as follows:

- 12 bytes, internal use
- 1 byte, card type
  - `0xff` missing
  - `0x01` downstream
- SFP data, 256 bytes per SFP, 16 SFPs per card
  - 128 bytes, SFP ID data
  - 128 bytes, SFP diagnostic data

17564 – Switch Connection Status Broadcast

- 32-bits, HDxRouter's primary IP address
- 16-bits, 0 for the Upper (only) HDxRouter chassis
- 16-bits, first port number in the list (list starts from 1, not 0)
- 16-bits, last port number in the list
- N * 16-bits, the input port that is connected to output port X, zero means the port is NOT connected

- Example: `c0 a8 0d 19  00 00  00 01  02 40  00 11  00 12…`
  IP address = 192.168.13.25
  Chassis = 0
  First Port number = 1
  Last Port number = 576 (`0x0240`)
  Output 1 is connected to input 17 (`0x0011`)
  Output 2 is connected to input 18 (`0x0012`)

17565 – Port Settings (localhost only)

576 sets of the following data:

- (8 bits) output level 0 is off
- (8 bits) input level
- (16 bits) connected to input port number, starting from 0, big-endian format

A 'C' data structure for this would be:

```
struct s_block {
        unsigned char  output_level;
        unsigned char  input_level;
        unsigned short        connectedTo;  /* assumes a short is 16 bits */
} data_block [576];
```

17566 – Hardware Status

- Reserved for future use.

17600 – System Data

- System data consists of 18 bytes. The first two bytes have the current alarm contact settings (big-endian). Each bit corresponds to a contact. Bit 0 (lsb) of byte 1 is the bit for the Power Supply 1 alarm, and bit 7 (msb) is the bit for the 'any' alarm. The next five bytes are for the installed card inventory. A '0' bit means the card is installed and a '1' bit means the card is removed. The other bytes are for internal use.

	Alarms		Inventory				
Bit	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
0	Alarm 9	Alarm 1	Card 1	Card 9	Card 26	Card 14	Card 31
1	Alarm 10	Alarm 2	Card 2	Card 19	Card 27	Card 15	Card 32
2		Alarm 3	Card 3	Card 20		Card 16	Card 33
3		Alarm 4	Card 4	Card 21		Crad 17	Card 34
4		Alarm 5	Card 5	Card 22	Card 10	Card 18	Card 35
5		Alarm 6	Card 6	Card 23	Card 11	Card 28	Card 36
6		Alarm 7	Card 7	Card 24	Card 12	Card 29	
7		Alarm 8	Card 8	Card 25	Card 13	Card 30	

Table 5: Hx576 System Data

17601:17636 – Card Data

- Each I/O card in the HDxRouter has an associated TCP port. Card 1 uses port 17601, card 2 uses 17602, and so on. The data is formatted as follows:

- 12 bytes, internal use
- 1 byte, card type
  - `0xff` missing
  - `0x01` downstream
- SFP data, 256 bytes per SFP, 16 SFPs per card
  - 128 bytes, SFP ID data
  - 128 bytes, SFP diagnostic data

17564 – Switch Connection Status Broadcast

- 32-bits, MxRouter's primary IP address
- 16-bits, 0 for the Upper (only) router chassis
- 16-bits, first port number in the list (list starts from 1, not 0)
- 16-bits, last port number in the list
- N * 16-bits, the input port that is connected to output port X, zero means the port is NOT connected

Example: `c0 a8 0d 19  00 00  00 01  00 30  00 11  00 12…`

IP address = 192.168.13.25
Chassis = 0
First Port number = 1
Last Port number = 48 (`0x0030`)
Output 1 is connected to input 17 (`0x0011`)
Output 2 is connected to input 18 (`0x0012`)

17565 – Port Settings (localhost only)

48 sets of the following data:

- (8 bits) output level 0 is off
- (8 bits) input level
- (8 bits) connected to input port number, starting from 0

A 'C' data structure for this would be:

```
struct s_block {
        unsigned char output_level;
        unsigned char input_level;
        unsigned char connectedTo;
} data_block [48];
```

17566 – Hardware Status

Reserved for future use.

17600 – System Data

System data consists of 6 bytes. The first byte has the current alarm contact settings. Each bit corresponds to a contact. Bit 0 (lsb) is the bit for the Power Supply alarm, and bit 3 is the bit for the 'any' alarm. The next byte is for the installed card inventory. A '0' bit means the card is installed and a '1' bit means the card is removed. The other bytes are for internal use.

	Alarms	Inventory
	Byte	Byte
Bit	0	1
0	Alarm 1	Card 1
1	Alarm 2	Card 2
2	Alarm 3	Card 3
3	Alarm 4	
4		
5		
6		
7		

Table 6: MX48 System Data

17601:17603 – Card Data

Each I/O card in the router has an associated TCP port. Card 1 uses port 17601, card 2 uses 17602, and so on. The data is formatted as follows:

- 6 bytes, internal use
- 1 byte, card type
  - `0xff` missing
  - `0x02` normal
  - `0x05` reclocked
- SFP data, 256 bytes per SFP, 16 SFPs per card
  - 128 bytes, SFP ID data
  - 128 bytes, SFP diagnostic data

17564 – Switch Connection Status Broadcast

- 32-bits, VxRouter's primary IP address
- 16-bits, 0 for the Upper (only) VxRouter chassis
- 16-bits, first port number in the list (list starts from 1, not 0)
- 16-bits, last port number in the list
- N * 16-bits, the input port that is connected to output port X, zero means the port is NOT connected

Example: `c0 a8 0d 19  00 00  00 01  02 80  00 11  00 12…`

IP address = 192.168.13.25
Chassis = 0
First Port number = 1
Last Port number = 640 (`0x0280`)
Output 1 is connected to input 17 (`0x0011`)
Output 2 is connected to input 18 (`0x0012`)

17565 – Port Settings (localhost only)

640 sets of the following data:

- (8 bits) output level   0 is off
- (8 bits) input level
- (16 bits) connected to input port number, starting from 0, big-endian format

A 'C' data structure for this would be:

```
struct s_block {
        unsigned char   output_level;
        unsigned char   input_level;
        unsigned short          connectedTo;  /* assumes a short is 16 bits */
} data_block [640];
```

17566 – Hardware Status

Reserved for future use.

17600 – System Data

System data consists of 28 bytes. The first two bytes have the current alarm contact settings (big-endian). Each bit corresponds to a contact. Bit 0 (lsb) of byte 1 is the bit for the Power Supply 1 alarm, and bit 7 (msb) is the bit for the 'any' alarm. The next five bytes are for the installed card inventory. A '0' bit means the card is installed and a '1' bit means the card is removed. The other bytes are for internal use.

	Alarms		Inventory				
Bit	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
0	Alarm 9	Alarm 1	Card 17	Card 1	Card 18	Card 2	Switch 1
1	Alarm 10	Alarm 2	Card 19	Card 3	Card 20	Card 4	Switch 1
2		Alarm 3	Card 21	Card 5	Card 22	Card 6	Switch 3
3		Alarm 4	Card 23	Card 7	Card 24	Card 8	Switch 4
4		Alarm 5	Card 25	Card 9	Card 26	Card 10	Switch 5
5		Alarm 6	Card 27	Card 11	Card 28	Card 12	Switch 6
6		Alarm 7	Card 29	Card 13	Card 30	Card 14	Switch 7
7		Alarm 8	Card 31	Card 15	Card 32	Card 16	Switch 8

Table 7: Vx640Video System Data

17601:17620 – Card Data

Each I/O card in the VxRouter has an associated TCP port. Card 1 uses port 17601, card 2 uses 17602, and so on. The data is formatted as follows:

- 16 bytes, internal use
- 2 bytes, card type
  - `0xff, 0xff` missing
  - `0x00, 0x62` VX640 I/O card (bits 7-4, ignore bits 3-0)
- signed 16-bits, card temperature (little-endian, bits 15-7)
  - example: 0x80 0x1d -> 0x1d80 -> 29.5°C
- 5 bytes, internal use
- SFP data, 256 bytes per SFP, 16 SFPs per card
  - 128 bytes, SFP ID data
  - 128 bytes, SFP diagnostic data

Service Request (LOS) Signaling

Some VX, MX and HDX routers have an optional mode that can broadcast a message over the network when a remote receiver requests attention. When enabled, the receiver will toggle its' TX pin, causing a LOS (loss of signal) event to occur at the router. When detected, the router broadcasts an 'Attention Request' message over the network. This message contains the IP address of the router and the port number that detected the request. It is left up to an external system to detect, decode and act upon this request.

17560 – Attention Request (UDP)

- 32-bits, router's primary IP address
- 16-bits, 0 for the Upper (or only) router chassis
  1 for the Lower router chassis
- 16-bits, port number (list starts from 1, not 0)
- 16-bits, key code in bits 0 – 7, 0x00 and 0xff are ignored

Example: `c0 a8 0d 19  00 00  00 06  00 55`
IP address = 192.168.13.25
Chassis = 0
Port number = 6
key code 85, 0x55

SNMP

The router can be monitored via SNMP. The MIB definition files can be downloaded from our ftp site and are also located on the router in the directory *`/usr/share/snmp/mibs/`*. The SNMP program may be configured by modifying the file *`/etc/snmp/snmpd.conf`*. Configuring SNMP is beyond the scope of this document, but many excellent sites on the web exist that explain configuration. One such site is: http://www.net-snmp.org/docs/man/snmpd.conf.html.

There are a few entries in the SNMP configuration file that you may want to modify:

- syslocation   Text to aid in locating the unit
- syscontact   Name/Phone/email address of a contact
- trap2sink   IP address/name of a system to receive SNMP traps

  The default name used is 'snmp.trap' and is listed in *`/etc/hosts`* as pointing to 127.0.0.1. The easiest method to set the trap address is to modify the *`/etc/hosts`* entry and not change the trap2sink setting in the configuration file.

NTP

NTP, or Network Time Protocol, is used to keep the internal clock on the router set to the correct time. All internal clocks drift over time, some slow, some fast. By enabling this service, you keep the router's clock synchronized to an external time reference. In order to use the service you must have a time server running on your network or can allow the router to access a public time server. NTP configuration is contained in the file */etc/ntp.conf*. Details about NTP configuration can be found at http://tldp.org/LDP/sag/html/basic-ntp-config.html.

System Log Files

The router records system events into files located in the directory */var/log/*. This directory is in RAM, so its' contents **are volatile**. If you wish to preserve log files, you may redirect the logs to another system. This is a standard feature of Linux's syslog program. This redirection is enabled by entries in the syslog configuration file */etc/syslog.conf*. Details about syslog.conf entries may be found at http://linux.about.com/od/commands/l/blcmdl5_syslogc.htm.

The two log files with the most interesting information are: */var/log/messages* and */var/log/api*. The former contains system events and the latter contains switch events and API commands and responses. More information about the API log file can be found in: **Configuring-the-ASCII-Interface.pdf**.

Below is an example */etc/syslog.conf* file that will send log messages to a remote server named 'extLogger'. 'extLogger' may be replaced with the IP address of the remote server.

```
*.notice;local4.none            /var/log/messages
local4.notice                   /var/log/api
*.crit                          /var/critical

#send log messages to external server named extLogger
*.notice                        @extLogger
*.notice                       @192.168.13.9
```

Linux systems are able to receive external syslog messages (from the router). Please refer to the documentation for your version of Linux to enable the logging of remote messages.

Windows will require third-party software to be able to receive syslog messages.

SD Firmware Version

There is a text file on the routers' internal SD card that contains the firmware version. This file is: */etc/vxr-release*. Example contents are:

- Vx80 v4.13
- Vx320-video v4.10
- MX48 v4.18
- Vx640 v4.14

Quick Reference

IP addresses:

• Primary	192.168.13.15 and 192.168.13.115
• Backup CPU	192.168.13.16

LOS Attention Broadcast port:	17560 (UDP)
Connection Broadcast port:	17564 (UDP)
I/O Port status:	17565 (UDP, localhost only)
Hardware sensor status:	17566 (TCP)
ASCII Command port:	17567 (TCP)
System status:	17600 (TCP)
I/O card status:	17601:17640 (TCP)

router MIB files:

- *LSI-ROOT.mib*
- *LSI-ROUTER-API-INTERFACE.mib*
- *LSI-SFP.mib*
- *LSI-VXROUTER.mib*

Important files or directories on the router

- */etc/hosts*
- */etc/inittab*
- */etc/ntp.conf*
- */etc/snmp/snmp.conf*
- */etc/syslog.conf*
- */etc/vxr-release*
- */var/log/api*
- */var/log/messages*
- */usr/share/snmp/mibs/*

Glossary

big-endian

The most-significant byte of a multi-byte value is stored first, followed by lesser significant bytes, and ending with the least-significant byte.

Example: `0x12345678` is stored as 12 34 36 78.

little-endian

The least-significant byte of a multi-byte value is stored first, followed by more significant bytes, and ending with the most-significant byte. This is the format used by x86 processor family.

Example: `0x12345678` is stored as 78 56 34 12.

MIB

Management Information Base – Used in conjunction with SNMP, a MIB is a database definition used to define management entities on a network. The MIB file is written using a subset of Abstract Syntax Notation One defined in RFC 2578. While readable by humans, it is typically compiled into a network management software package.

NTP

Network Time Protocol – a means for synchronizing internal clocks on networked devices.

SNMP

Simple Network Management Protocol – a standard IP protocol for monitoring and managing networked devices.