Sniffer Feature

Teldat-Dm 778-I

Legal Notice

Warranty

This publication is subject to change.

Teldat offers no warranty whatsoever for information contained in this manual.

Teldat is not liable for any direct, indirect, collateral, consequential or any other damage connected to the delivery, supply or use of this manual.

Table of Contents

I Related Documents

Teldat-Dm 775-I VRF Lite Feature

Chapter 1 Introduction

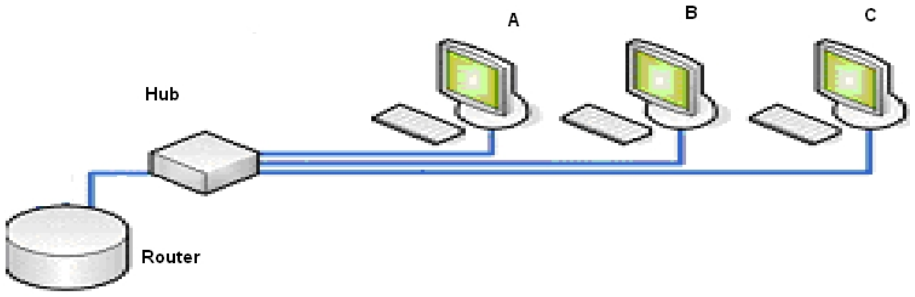
1.1 Sniffer Feature Introduction

In most networks, due to the shared nature of the transmission medium, a message sent to one particular device can be intercepted by another. Although in practice devices tend to ignore messages intended for others, they may decide not to ignore them and thus gain access to information traveling on another device's network.

A Sniffer is a program that can capture and log traffic from a network or network segment. Once it captures a packet, it can decode and analyze it according to the relevant RFC or standard.

This kind of network monitoring is especially useful for detecting bottlenecks and diagnosing other network issues.

In Ethernet environments for example, each device has its own uniquely identifying physical address called a MAC address (for Media Access Control). The link layer inserts the MAC address of the device upon transmission and checks it upon reception. If the packet's destination MAC address matches the address of the device, the frame is accepted and forwarded to the upper layers of the destination device for processing. To capture all packets, the Sniffer 'sets' the device's network card to a state known as 'promiscuous mode', enabling it to capture all frames traveling over the medium, even if the destination address does not match the device's address. In this way, it can capture information from any device connected to its network.



Traditional Ethernet Network

In Figure 1, packets that have device **A** as their source or destination address are received by all devices on the same broadcast domain, in this case **B** and **C**. Usually, both **B** and **C** drop the packets if the destination does not match their own MAC addresses. However, either one can intercept all traffic from station **A** simply by setting their network card to promiscuous mode.

1.2 Sniffer Feature: General Overview

The Sniffer feature allows you to capture traffic from the network the device is connected to and save the collected information in a file with a .CAP extension. This file can be extracted from the device's memory and used at a later time to analyze network behavior.

The following sections describe how the Sniffer feature works.

1.2.1 Capture File

By default, the sniffed traffic information is saved in the file named CAPTURE.CAP, though the user can assign a different name.

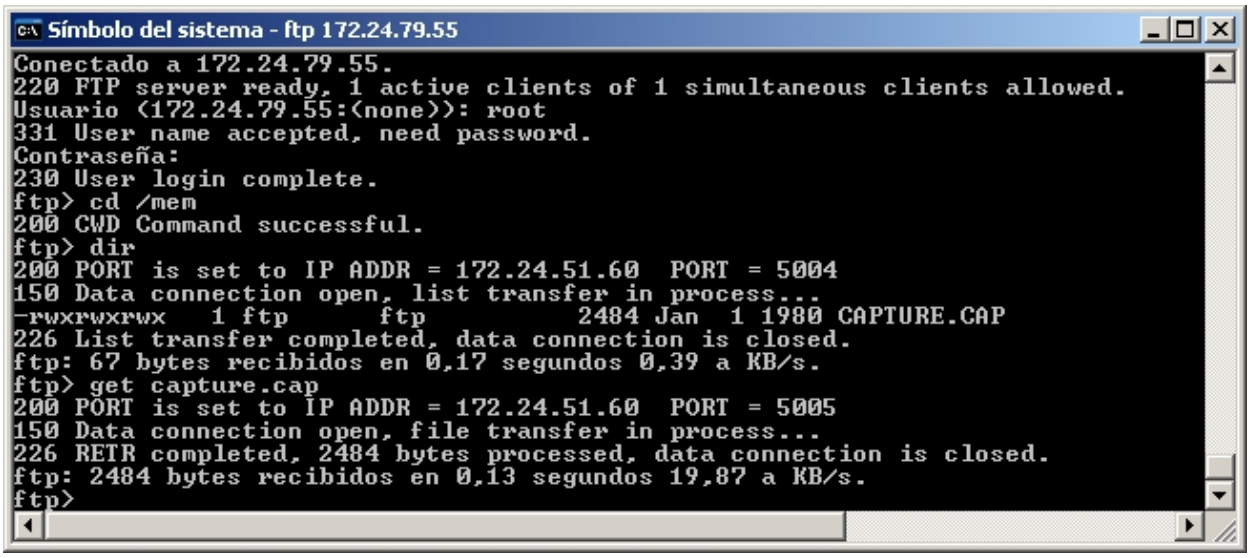
Example:

```
SNIFFER config>filename prueba
SNIFFER config>
```

In our example, the captured packets will be saved in the file named 'prueba.cap'. It is important to remember to save the configuration and restart the device so that the association takes effect.

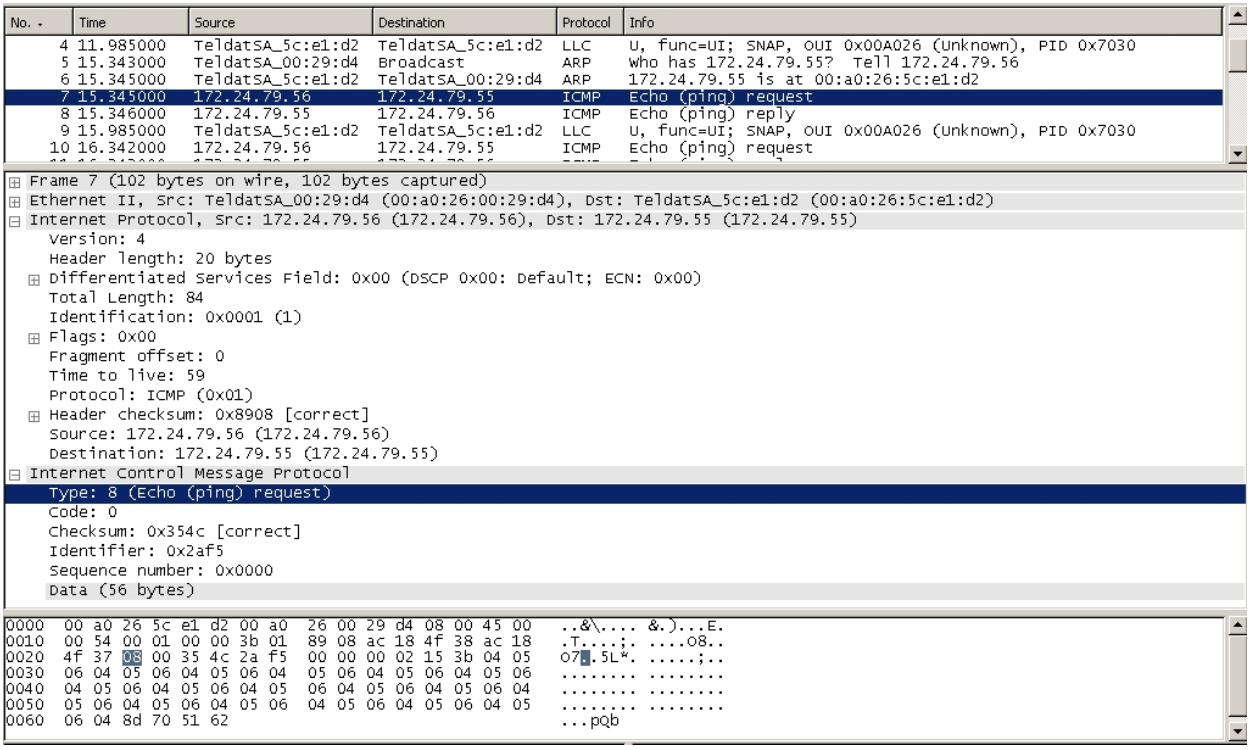
The 'CAP' file is stored in the device's memory / **mem** directory and can be accessed through an FTP connection.

Example:



Establishing an FTP connection and extracting the capture file

In the example in Figure 2, after the login process, an FTP connection has been established with the capture device, the contents of the /**mem** directory have been obtained and the CAPTURE.CAP file has been extracted using the '*get*' command. The file's contents are shown in Figure 3 using a protocol analyzer.



Displaying the CAPTURE.CAP file through a protocol analyzer

The analyzer used divides the screen into three sub-windows. The first sub-window displays the various captured packets; the second shows the packet highlighted in the first window in a more detailed form, identifying the different protocols involved; and the last one displays the byte-level data contained in the selected packet.

The packet flow corresponds to a ping process between device 172.24.79.56 and device 172.24.79.55, with the initial ARP message exchange shown first followed by the ICMP Request and Replay packet sequence.

1.2.2 Capture Modes

Devices connected to a network are constantly 'listening' to the medium waiting to receive data, checking the packet destination address to see if it matches their own and, if it does, accepting and processing it.

Based on this operating principle, the Sniffer feature offers two modes of packet capture:

· *single-host*: The device saves any packet with a source or destination address matching its own to the capture file.

· *promiscuous*: In this mode, the device captures and saves all packets circulating in the network that it is connected to.

1.2.3 Capture Device

There are three different ways to make a capture: specify a specific interface through which to capture traffic, e.g., *ethernet0/0*; select an '*any device*' capture, which permits you to capture traffic from the different interfaces active in the device, or select an ip-forwarder capture, which intercepts traffic at the IP layer when the packet is being routed.

Execution of a capture is only possible in the P 5 process, since it must be done on an interface that is already operative in the device.

Example:

```
*p 5
Config$ feature sniffer

-- SNIFFER configuration --
SNIFFER config$capture ethernet0/0 100 promiscuous
Capturing traffic (Press any key to abort)...
Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

Example:

```
-- SNIFFER configuration --
SNIFFER config$capture bri0/0 10 single-host
pcap_open_live failed: Chosen device not supported
CLI Error: Command error
SNIFFER config$
```

The first example shows that a capture of 100 packets has been executed on the ethernet0/0 interface using the network card in promiscuous mode. The second shows that an interface without the Sniffer feature enabled has been selected and the console displays the following error message: '*pcap_open_live failed: Chosen device not supported* '.

In '*any device'* captures, packets are directly obtained from the kernel and encapsulated in a 'pseudo-protocol' called SLL, which adds a dummy layer 2 header to them.

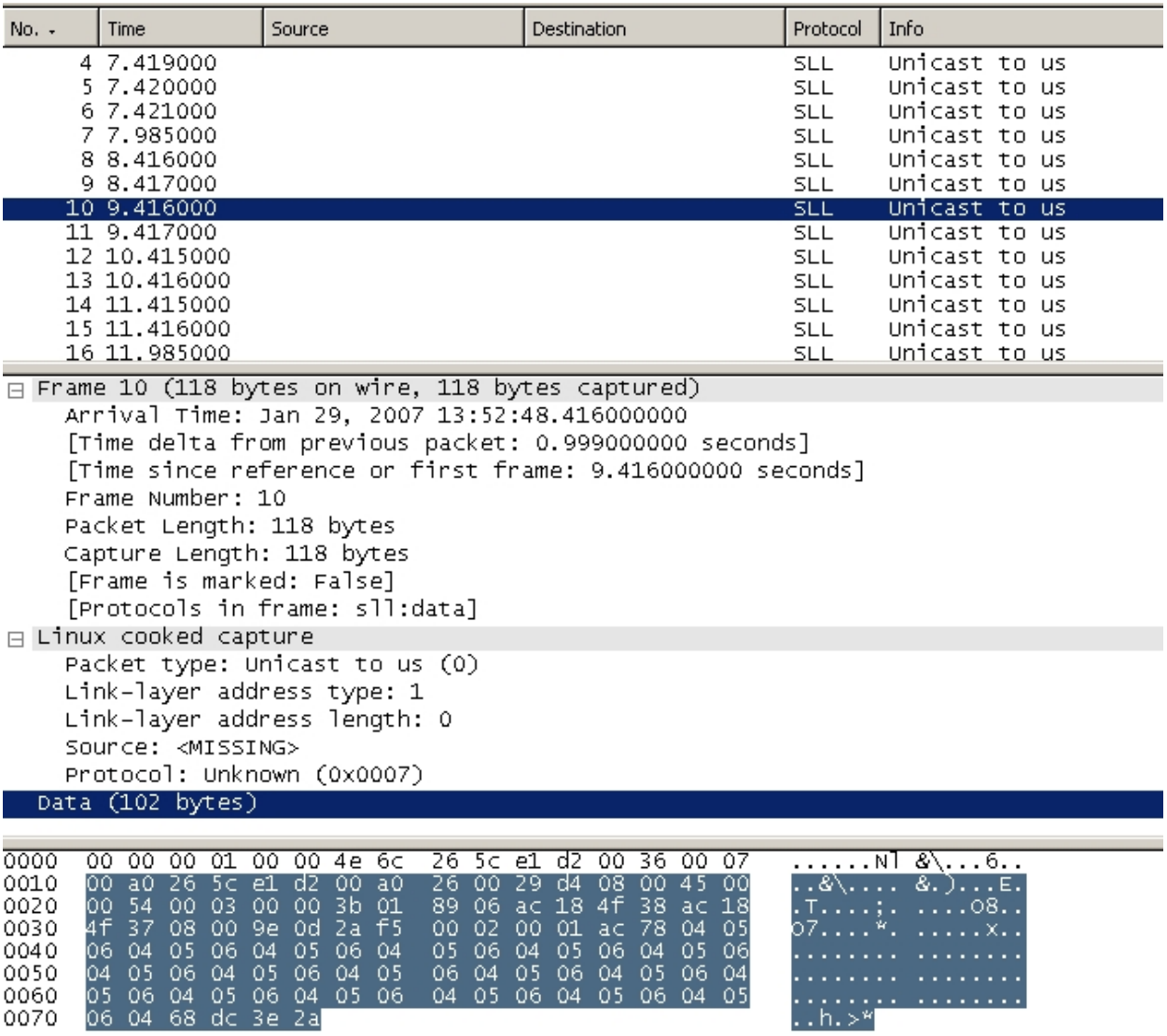
Example:

```
-- SNIFFER configuration --
SNIFFER config$capture any 100 single-host
Capturing traffic (Press any key to abort)...

Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

In this example, an 'any device' capture of 100 packets has been launched in 'single-host' mode with the aim of monitoring the same Ping process shown in Figure 3. Figure 4 shows the captured packet sequence. Unlike the first capture, there is no detailed analysis of the selected packet identifying the different protocols that it is composed of. Instead, we have a payload of 102 bytes encapsulated in SLL.

The packet corresponds to an ICMP Echo Request and an analysis of the data field would allow you to identify the Ethernet, IP, ICMP headers and finally the real user data.



Displaying an 'any device' capture in Figure 3's ping process

Example:

```
-- SNIFFER configuration --
SNIFFER config$capture any 100 promiscuous
Warning: Promiscuous mode not supported on the "any" device
Warning: Switching capture to single host...
Capturing traffic (Press any key to abort)...
Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

This last example reveals the incompatibility between promiscuous mode and *any device* captures. If you try and force this combination, the device displays a warning message indicating that the capture mode has been changed internally to 'single-host'.

Example:

```
-- SNIFFER configuration --
SNIFFER config$capture ip-forwarder 100
Capturing traffic (Press any key to abort)...
Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

This example captures 100 packets from IP traffic being routed by the device.

Chapter 2 Configuration

2.1 Sniffer Feature Configuration

The following example shows you how to access the Sniffer feature configuration menu.

Example:

```
*p 4
Config>feature sniffer
-- SNIFFER configuration --
SNIFFER config>
```

The available commands are:

Command	Function
? (HELP)	Displays the available commands or their options.
FILENAME	Allows you to specify the filename where the captured packets are to be stored.
MAX-SIZE	Configures the maximum size of the file resulting from the capture.
EXIT	Returns to the configuration prompt (Config>).

If you access the configuration menu from the RUNNING_CONFIG (P5), the available commands are:

Command	Function
? (HELP)	Displays the available commands or their options.
CAPTURE	Start of packet capture.
FILENAME	Allows you to specify the filename where the captured packets are to be stored.
MAX-SIZE	Configures the maximum size of the file resulting from the capture.
EXIT	Returns to the configuration prompt (Config>).

Example:

```
*p 5
Config$feature sniffer
-- SNIFFER configuration --
SNIFFER config$?
  capture     Start capturing packets
  filename    Choose default file name
  max-size    Maximum capture file size (KBytes)
  exit        Exit to parent menu
SNIFFER config$
```

The discrepancy in the Sniffer feature menu between the ‘CONFIG’ and ‘RUNNING-CONFIG’ processes is due to the fact that the captures must be made on interfaces that are already running in the device, which limits the use of the *capture* command to P 5.

2.1.1 ? (HELP)

Displays the list of available commands.

From P 4:

Syntax:

```
SNIFFER config>?
```

Example:

```
SNIFFER config>?
  filename    Choose default file name
  exit        Exit to parent menu
SNIFFER config>
```

From P 5:

Syntax:

```
SNIFFER config$?
```

Example:

```
SNIFFER config$?
  capture     Start capturing packets
  filename    Choose default file name
  exit        Exit to parent menu
SNIFFER config$
```

2.1.2 CAPTURE

Executing this command initiates packet capture in the selected interface. You also need to specify the number of packets to be captured and the operating mode.

The capture command is only available in the P 5 process in the Sniffer feature menu.

Syntax:

```
SNIFFER config$capture ?
  <interface>    Interface name
  any            Any device
  ip-forwarder   Any IP forwarded packet
SNIFFER config$
```

2.1.2.1 <interface> capture

Allows you to specify a specific interface on which to capture packets. If the selected interface does not allow capturing traffic, the console displays an error message.

Syntax:

```
SNIFFER config$capture <interface> <No. of packets(0..100000)> <mode>
```

Example:

```
SNIFFER config$capture ethernet0/0 200 promiscuous
Capturing traffic (Press any key to abort)...

Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

Example:

```
SNIFFER config$capture ethernet0/0 100 single-host
Capturing traffic (Press any key to abort)...

Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

The first example shows that a capture of 200 packets has been executed in promiscuous mode, and the second that a capture of 100 packets has been made in ‘single-host’ mode. In the first case, all packets that have reached the Ethernet network card are available, while in the second case, only those packets with a source or destination address matching that of the device's network card are captured.

The user can abort the capture process simply by pressing any key. The .CAP file collects the packets captured up to that moment.

2.1.2.2 any capture

Makes an ‘any device’ capture. That is, packets that enter or leave the device encapsulated in a dummy layer 2 frame are available. The data field is made up of the real layer 3 packet and, if possible, the real link layer header.

Syntax:

```
SNIFFER config$capture any <No. of packets(0..100000)> <mode>
```

Example:

```
SNIFFER config$capture any 10 single-host
Capturing traffic (Press any key to abort)...

Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

2.1.2.3 ip-forwarder capture

This allows you to capture packets being routed by the router. You must specify the number of packets that need to be captured. You can select which VRF to use (see manual Teldat-DM775-I VRF Lite Feature), but if you don't select one, the main VRF is used. You can also select an access list to select which traffic to capture; if you don't select an access list, then all routed traffic is captured.

Syntax:

```
SNIFFER config$capture ip-forwarder <No. of packets(0..100000)>
   access-list    Capture packets that match access-list
      <1..9999>      Value in the specified range
   vrf            Capture packets on VRF
      <word>         Text
   <cr>
```

Example:

```
SNIFFER config$capture ip-forwarder 1000
Capturing traffic (Press any key to abort)...

Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

Example:

```
SNIFFER config$capture ip-forwarder 1000 access-list 101 vrf VRF1
Capturing traffic (Press any key to abort)...

Capture finished
  CAP file is available, by means of FTP connection,
   in /mem directory.
SNIFFER config$
```

The first example shows that a capture of 1000 packets being routed by the device has been executed. The second example specifies a filter through access-list 101 so that only those packets that match the said access-list are captured. This also specifies that the capture is executed in the VRF1 entity instead of the main VRF.

2.1.3 MAX-SIZE

Configures the maximum size of the file resulting from the **capture** command being executed. This size is configured in Kbytes and the default is 1024. If this size is exceeded during a capture, the capture automatically stops. Bearing in mind that the file is saved in the device RAM memory, if you are going to execute captures that occupy a great deal of memory it's a good idea to check and see if there is enough free memory by executing the **memory** command found in the monitoring menu.

Syntax:

```
SNIFFER config$max-size <1..8096>
```

Example:

```
SNIFFER config$max-size 100
```

2.1.4 EXIT

Exits the Sniffer feature configuration environment and returns to the previous configuration prompt.

Syntax:

```
SNIFFER config>exit
```

Example:

```
-- SNIFFER configuration --
SNIFFER config>exit
Config>
```