3 Capturing Application Traffic: Overview

The first step to studying an application in AppTransaction Xpert is to record an application transaction in one or more packet traces. Packet traces (also known as *capture files*) form the raw data for creating Transaction Analyzer models. This section includes general information about how to capture traffic and lists the supported capture file formats.

AppTransaction Xpert includes an application capture utility that enables you to record traffic throughout your network. You can install an unlimited number of capture agents and manage these agents from your local computer.

For more information, see

- General Advice for Capturing Application Traffic
- Capturing an Application from Multiple Locations
- Capturing Traffic with Other Programs
- Capturing Multi-Tier Transactions with Other Programs
- Capturing HTTPS Transactions for Automatic SSL Decryption

Related Topics

- *Capturing Traffic with AppTransaction Xpert*
- *Capturing an Application from Multiple Locations*
- *Supported Packet Trace Formats*

General Advice for Capturing Application Traffic

AppTransaction Xpert includes an application capture utility (described in Capturing Traffic with AppTransaction Xpert) that enables you to capture traffic. Alternately, you can use a number of third-party tools to capture traffic. These tools are generally called *network analyzers*, although certain RMON probes can also capture traffic. (This section refers to both kinds of tools as *analyzers* for the sake of simplicity.) AppTransaction Xpert can import capture data from both .enc (uncompressed) and .csv formats. You can also assign preprocessor scripts to convert packet traces automatically in other formats during the import process, as described in Supported Packet Trace Formats.

Analyzer-specific information is beyond the scope of this documentation, as individual equipment can vary. However, the following are guidelines for capturing traffic for analysis by AppTransaction Xpert:

1) Configure the analyzer's filtering capabilities to focus on hosts of interest. Filtering traffic is not imperative because the Trace File Filter operation enables you to filter on specific hosts. However, filtering keeps the packet traces smaller and enables the analyzer to accommodate a longer duration for the capture, given limited memory and disk resources.

2) Aim for a capture that is as *clean* (that is, clear of irrelevant traffic) as possible. It may be best to perform captures when network traffic is minimal, such as late in the evening or during the weekend, to exclude irrelevant packets from the packet trace. In any event, traffic can be removed during and after the import process. For more information, see Filtering Traffic.

3) Place the analyzer as close as possible to the application instance(s) relevant to the study. For example, to study performance on a file server, capture on the same LAN as the server so that network delays across a WAN link or the Internet do not affect the analyzer's view of the relevant information (application performance on the file server).

4) Capture one user-level application in its entirety. For example, to study an FTP application, capture the exchange from the initial request on the client host to the conclusion of the download.

   Note—If an application includes user *think time* (for example, waiting for a user to click OK before proceeding), you must identify this think time in the resulting Transaction Analyzer model. This is especially important to ensure valid results and diagnoses from AppDoctor. For more information, see User Think Time.

Related Topics

- *Capturing Application Traffic: Overview*

Capturing an Application from Multiple Locations

You can capture an application exchange from different network locations and import the resulting packet traces. AppTransaction Xpert merges the capture data and creates a unified model of an application exchange. This enables you to model your application using traffic captured from multiple network locations.

Using the application-capture features, you can easily capture an application transaction from multiple locations. For more information, see Capturing Traffic with AppTransaction Xpert.

A multi-capture Transaction Analyzer model has the following advantages over a model created from a single packet trace:

- Greater realism and accuracy—A capture agent records packets as they travel to and from the host on which the capture agent is installed. It does *not* record activity (such as actual packet arrivals and departures) on other hosts. A single-capture task uses the specified bandwidth and latency values to estimate arrival and departure times on remote tiers.

  A multi-capture task includes traffic data from multiple hosts. Given multiple files captured simultaneously, AppTransaction Xpert calculates (rather than estimates) the actual packet transmission times between tiers. The result is a more realistic and accurate model of the application.

- Greater flexibility in modeling multi-tier applications—It is impossible to capture an entire multi-tier application from one host. This is especially true if hosts are located in different network segments, such as in a WAN. If you can capture and import traffic on multiple hosts, you can model virtually any application, regardless of the number of tiers.

When You Can Import Multiple Packet Traces

You can import multiple packet traces into one Transaction Analyzer model *only* if the capture agents recorded the same application exchange at the same time.

A multi-capture import is advantageous in the following circumstances:

- In a production environment in which the tiers are not collocated, as in a WAN
- In a complex multi-tier application exchange in which no single tier sees all traffic

Importing multiple packet traces is most advantageous when the capture agents are installed in different network segments. You can import packet traces that were captured on computers in the same high-bandwidth, low-latency network; there is no advantage to doing this, however, if all the traffic is visible in one packet trace.

AppTransaction Xpert can automatically match and merge packet trace data. Even if it cannot match traffic between two or more files, you can still import the data. However, you must synchronize the packet traces manually so that AppTransaction Xpert can determine a common time frame for all traffic. For more information, see Manual Merge (Synchronizing Packet Traces Manually).

AppTransaction Xpert can merge packet traces successfully even if there is Network Address Translation (NAT) between tiers. Additionally, AppTransaction Xpert handles most types of firewalls between tiers, including packet filter firewalls and stateful/stateless inspection firewalls. However, AppTransaction Xpert cannot merge packet traces automatically if there are some varieties of application proxy firewalls between tiers; in this event, you must synchronize the files manually (as described in Manual Merge (Synchronizing Packet Traces Manually)).

When You SHOULD NOT Import Multiple Packet Traces

You should not import multiple packet traces that

- Do not capture the same application exchange
- Were not created simultaneously

If you created multiple packet traces that came from the same high-bandwidth, low-latency network, where all of the traffic is visible in one packet trace, there is no advantage to importing multiple packet traces.

Capture Requirements

Remember the following when creating multiple packet traces:

- Capture all traffic simultaneously. If the captures are not simultaneous, the packet traces will contain different packets and AppTransaction Xpert will not be able to merge them.

  The following method is recommend:

  a) Start all capture agents simultaneously

  b) Run one application transaction while filtering out unrelated traffic

  c) Stop all capture agents simultaneously

- It is not necessary to synchronize the clocks for capture agents because the AppTransaction Xpert automatically synchronizes the packet traces. Therefore, you can merge packet traces that were captured in different time zones, as long as they were captured simultaneously.
- You must know the bandwidth between the different capture locations. This information is necessary to create an accurate model. (Because AppTransaction Xpert auto-detects the latency between capture locations, you do not need to specify this information.)

  Always specify the *minimum* bandwidth—that is, the bandwidth of the slowest link between capture locations. You can usually estimate the bandwidth based on the link type (56k, T1, etc.).

  If the you do not know the bandwidth measurements in your network, use PathProbe to estimate the information. For more information, see Measuring Network Connections with PathProbe. (PathProbe is not available in all solutions.)

Capturing Multi-Tier Applications

AppTransaction Xpert can only merge packet traces that have common traffic, where packets in one packet trace match those in another. For example, consider the following three-tier transaction, as shown in the following figure:

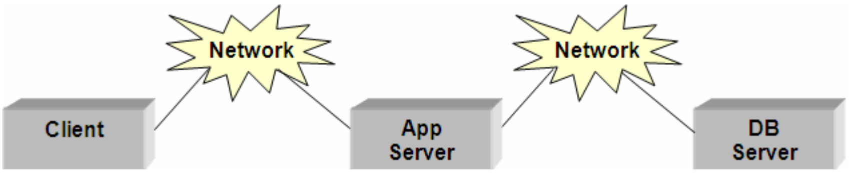
1) Client issues a request to the App Server

2) App Server forwards the request to the DB Server

3) DB server responds to the App Server

4) App Server forwards the response to the Client

Figure 3-1 Example Three-Tier Application



When determining where (that is, at which network locations) to capture this application, consider the following:

- Capturing only at the Client and the DB Server is not an ideal solution. Because the two captures would not contain any common packets, AppTransaction Xpert cannot merge the results automatically. If you cannot capture at the App Server tier, you could synchronize the packet traces manually (as described in Manual Merge (Synchronizing Packet Traces Manually)). However, this is less desirable than having AppTransaction Xpert merge the packet traces automatically.
- Capturing only at the Client and the App Server would work partially. Both captures would contain the packets from steps 1 and 4, so AppTransaction Xpert could merge them successfully. AppTransaction Xpert would also import the packets from steps 2 and 3 (captured at the App Server), but would not produce a merged model of these steps.
- Importing packet traces from hosts that are separated by a high-bandwidth, low-latency network provides no advantage over importing one packet trace. Assume that the client is separated by a WAN. If the App Server and the DB Server are separated by a high-bandwidth, low-latency network, the recommended method is to capture at the Client and the App Server only. If the App Server and the DB Server are separated by a MAN or WAN, the recommended method is to capture at all three tiers.

Capturing Applications on a Busy Network

Suppose you want to capture a multi-tier application on a busy network, in which a middle tier (such as the App Server in Figure 3-1) receives multiple client requests per second. There are several strategies to capture applications in such an environment:

1) Capture the application when no other client requests are hitting the servers. You can do this most reliably in a lab environment or during off-hours.

2) Configure a load balancer to create a temporary *clear channel* for your application. You might be in a production environment where a load balancer distributes requests from multiple web servers. Sometimes you can configure the load balancer temporarily to distribute requests to all servers except one (*rsv_server* in this example). You can then capture on *rsv_server* while only the test client hits this server's specific address.

   Depending on the load balancer, the configuration change might be as simple as changing the HTTP port on *rsv_server*. Many load balancers automatically determine that they cannot connect to *rsv_server* on the standard port and seamlessly distribute requests to the other web servers. The test client can then connect to *rsv_server* using the new port number.

   With this configuration, you can filter on *rsv_server*'s address when you capture the application. Suppose you capture a three-tier application, in which multiple load-balanced web servers distribute requests evenly to one or more database servers. To generate *clean* captures, you can filter on *web_client<—>rsv_server* and *rsv_server<—>database_server*.

   This is often the easiest way to capture cleanly in a production environment. Historically, AppTransaction Xpert users have been willing to use this temporary and potentially simple technique. One benefit is that the back-end tier receives other transaction requests during the capture. The primary drawback is that loads on the other servers might increase temporarily.

3) Filter on all tiers to obtain only the traffic associated with one client request. You can filter during the capture (on host, port, etc.), during the AppTransaction Xpert import, or within AppTransaction Xpert itself. This can be the most challenging of these four strategies.

4) Capture traffic between the first and second tiers to make sure that you see one client's traffic only. In this example, this would be the Client and the App Server. To do this, capture the traffic between the client and the tier that initially receives the client's request (app server, web server, etc.). When you import this traffic, AppTransaction Xpert categorizes the remaining tiers as part of the application delay at the second tier.

   This is often a good initial strategy; it enables you to identify any bottlenecks in the interaction between the first two tiers. If the largest component of delay is application delay at the second tier, then you can analyze the remaining tiers using one of the other capture methods.

You can combine these methods when you analyze a particular application. For example, the first method (capturing in a lab environment or during network down time), results in a *clean* end-to-end capture of the application; however, you do not see the effects of other users simultaneously hitting the middle tier. You might want to compare these results with the results from the last method (focusing on client-side traffic) to see if the middle tier's processing delay changes significantly.

Related Topics

- *Capturing Application Traffic: Overview*

Capturing Traffic with Other Programs

Supported Packet Trace Formats

For a list of packet traces that can be imported, see Supported Packet Trace Sources and Protocols.

Additionally, the following table lists supported formats from ProConvert (a conversion utility available from WildPackets, Inc.)

Table 3-1 File Formats Supported by ProConvert

Vendor	Product
AG Group	EtherPeek
Agilent Technologies	Agilent LAN Analyzer
Fluke	Protocol Inspector
HP/Agilent Technologies	Internet Advisor LAN (Win95/98) Internet Advisor LAN/49xx (Pre-Win95)
IBM	DatagLANce
Microsoft	Network Monitor (SMS 1.2 and 2.0, NT Server)
Network General	Sniffer (including compressed files) SnifferBasic/Pro (.cap format through release 3.5) For more information, see Capturing Traffic with Network General's Sniffer.
Network Instruments	Observer
Novell	LANalyzer for Windows
Precision Guesswork	LANWatch
Shomiti	Surveyer
Sun Micrososystems	snoop (free w/Solaris)
TTC	Fireberd500 Fireberd500 PC
Wavetek Wandel Goltermann	DA-30 (DOS version) Domino

Capturing Traffic with Network General's Sniffer

Remember the following guidelines when creating packet traces with Sniffer:

1) Capture trace data, not a "summary file". Sniffer and certain other analyzers provide summary information as an alternative way to present the capture data. In the case of Sniffer, this format is called "Expert Data", or the "Expert Analyzer Output File Format", and provides high-level information on the observed traffic, such as the number of connections that were opened between hosts, the amount of data transferred, etc. This type of information is not useful. Make sure to collect the packet-by-packet traffic information and not the summary form.

2) If you do not have a license for AppTransaction Xpert Decode Module, use ASCII Sniffer files (if possible) rather than binary files. You can import uncompressed binary (.enc, tcpdump) files; when you import such files, AppTransaction Xpert shows application-layer decodes in a combination of ASCII and hexadecimal. This works well for text-based protocols like HTTP, but less well for binary protocols like Sybase and Oracle. Network General's Sniffer includes a "Print to File" operation that enables you to create ASCII packet traces. When you import these packet traces into AppTransaction Xpert, you can view decodes for any application that Sniffer can decode.

Procedure 3-1   Creating an ASCII Packet Trace with Sniffer Pro for Windows

1 Open the trace file you want to import in Sniffer.

2 Make sure that the Decode tab at the bottom of the file window is selected.

3 Choose Display Setup from the Sniffer Display menu.

➥ The Display Setup dialog box appears.

4 Click the Summary Display tab and make sure that the Show all layers checkbox is checked. Then click OK.

5 Choose File > Print.

➥ The Print dialog box appears.

6 In the Print dialog box, make sure that:

- The Print to File checkbox is enabled
- The Summary and Detail checkboxes are enabled
- The Hexadecimal checkbox is cleared

7 Click OK to create the file.

End of Procedure 3-1

Procedure 3-2   Creating a Packet Trace with Sniffer for DOS

1 Set the Display options as follows:

- Display > Summary > All Layers = On
- Display > Summary > Relative Time = On
- Display > Summary > Bytes = On
- Display > Details = On
- Display > Name Width = 31
- Display > Expert = Off

2 Select Display > Print > Print to file and enter the file name.

End of Procedure 3-2

Related Topics

- *Capturing Application Traffic: Overview*

Capturing Multi-Tier Transactions with Other Programs

This section describes a workflow for capturing multi-tier transactions with external tools (such as Network General's Sniffer) and importing the resulting capture data. This workflow is useful if both of the following conditions are true:

- You want to capture a multi-tier transaction from multiple network locations.
- You must use a third-party protocol analyzer or sniffer tool, and cannot use AppTransaction Xpert capture agents.

Note—The AppTransaction Xpert capture utility is the preferable solution for capturing transactions from multiple locations, because you can start and stop all capture agents simultaneously from one location using Capture Manager (as described in Running a Capture). Therefore, use this workflow only if you must capture the transaction using external tools.

Workflow Description

The following steps outline the workflow for capturing transactions from multiple locations using third-party tools:

1) Start the sniffer or protocol analyzer.

2) Run one or more instances of the transaction, and send an ICMP ECHO message immediately before and after each instance.

   Immediately before the transaction of interest is run, the initiating host must send a “transaction-start” ICMP ECHO message to every host capturing the transaction. AppTransaction Xpert uses these messages as markers to determine when individual transactions start. The packet traces must include these markers for AppTransaction Xpert to extract the relevant capture data.

3) For each agent that will participate in the capture, make sure that the agent does not use the Default capture filter. The Default capture filter excludes ICMP traffic. (You can create a modified version of the Default filter that does not filter out ICMP traffic, which you can then use for this type of capture operation.)

4) Start the capture process on every host participating in the capture.

5) Run one or more instances of the transaction of interest, so that each transaction run (including the “transaction-start” message) is included in the resulting capture data.

6) Download all packet traces to the computer running AppTransaction Xpert.

7) Create transaction-specific packet traces, as described in Extracting Transaction Data from One or More Packet Traces.

   AppTransaction Xpert includes a Packet Trace File Slicer utility that reads the raw capture data and “slices” the data into separate transaction-specific packet traces that you can import. For each transaction, the slicer creates a set of packet traces (one per capture location) and places them into a separate subdirectory.

8) Optionally, import the transaction-specific packet traces using Batch Analyzer.

   Batch Analyzer is useful for this workflow because you can import data for multiple transactions and create multiple Transaction Analyzer models in one operation. For more information, see Batch Analyzer.

Extracting Transaction Data from One or More Packet Traces

Procedure 3-3 describes how to use the Transaction Extractor utility to generate packet traces. This utility is intended for specific inputs and outputs.

Input Requirements

The Transaction Extractor takes as inputs one or more packet traces. The packet traces often include irrelevant traffic. Note the following requirements:

1) All packet traces in the operation must be roughly concurrent—that is, they must capture the same time window and the same set of transactions.

2) Before launching a transaction, the initiating tier must send an ICMP ECHO message to all tiers that will participate in the transaction. The Transaction Extractor uses these messages to calculate the start of each transaction.

Procedure 3-3 Extracting Transaction Data from non-Packet Trace Files

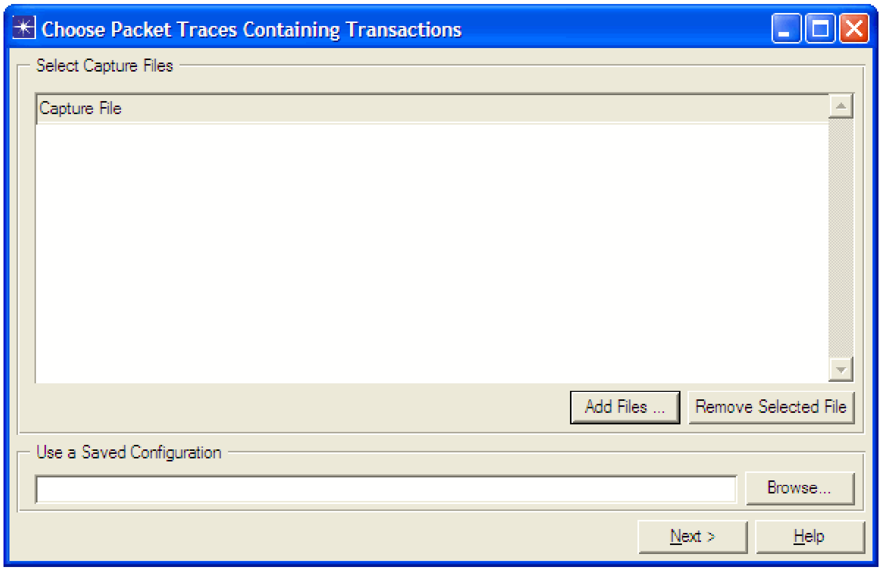
1 Capture the transaction(s) of interest as described in Workflow Description.

2 Open an application task () file, if one is not already open.

3 Choose Advanced > Extract Transactions using Pings…

➥ The Choose Packet Traces Containing Transactions dialog box appears.

Figure 3-2 “Choose Packet Traces Containing Transactions” Dialog Box



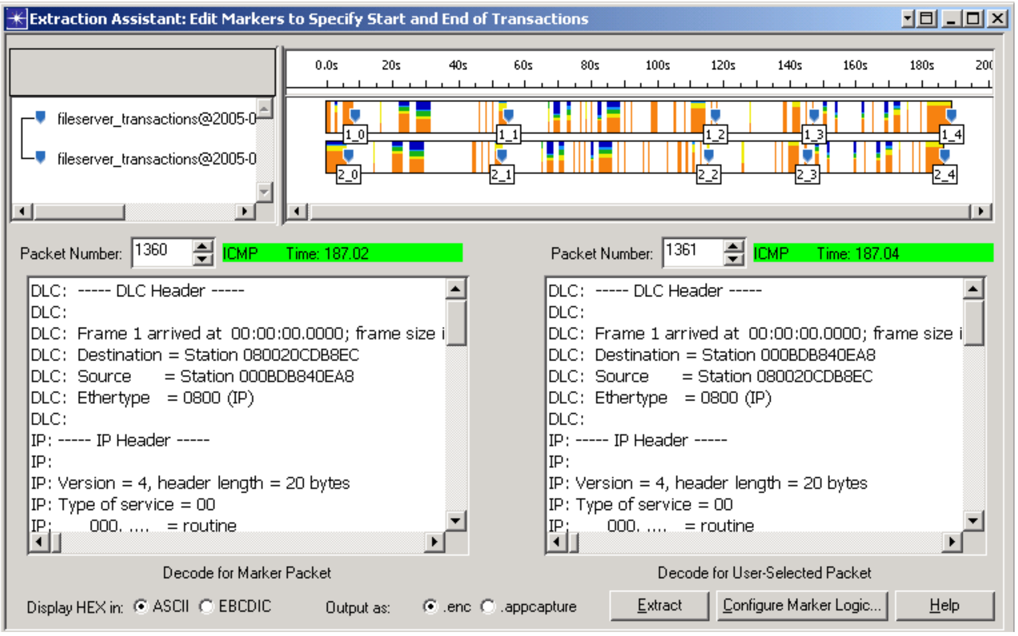
4 Click Add Files… and select one or more packet traces that contain transactions of interest. The selected set of files must meet the requirements specified in Input Requirements.

Note—When you complete an extraction, AppTransaction Xpert saves a settings file (settings.txt) in the output directory for the operation. If you want to repeat a previous operation, click the Browse… button next to the “Use a Saved Configuration” field, browse to the output directory of the previous operation, and select the settings.txt file.

5 Click Next.

➥ The Extraction Assistant dialog box appears. The timeline pane (upper-right) graphs the traffic in the packet traces you selected in step 4. This graph shows markers that specify the *estimated* transaction start and end times.

Figure 3-3 "Extraction Assistant" Dialog Box


Extraction Assistant: Edit Markers to Specify Start and End of Transactions
0.0s
20s
40s
60s
80s
100s
120s
140s
160s
180s
200
fileserver_transactions@2005-0
fileserver_transactions@2005-0
1_0
1_1
1_2
1_3
1_4
2_0
2_1
2_2
2_3
2_4

Packet Number: 1360 ICMP Time: 187.02

```
DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 1 arrived at  00:00:00.0000; frame size i
DLC:  Destination = Station 080020CDB8EC
DLC:  Source      = Station 000BDB840EA8
DLC:  Ethertype   = 0800 (IP)
DLC:
IP: ----- IP Header -----
IP:
IP: Version = 4, header length = 20 bytes
IP: Type of service = 00
IP:       000. ....   = routine
```

Decode for Marker Packet

Packet Number: 1361 ICMP Time: 187.04

```
DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 1 arrived at  00:00:00.0000; frame size i
DLC:  Destination = Station 000BDB840EA8
DLC:  Source      = Station 080020CDB8EC
DLC:  Ethertype   = 0800 (IP)
DLC:
IP: ----- IP Header -----
IP:
IP: Version = 4, header length = 20 bytes
IP: Type of service = 00
IP:       000. ....   = routine
```

Decode for User-Selected Packet

Display HEX in: ● ASCII ○ EBCDIC   Output as: ● .enc ○ .appcapture

6 Click the Configure Marker Logic… button, and enter the following values:

- The IP address of the host that generated the ICMP ECHO messages at the start and end of the transactions.
- The minimum time between markers, which determines whether a series of pings is seen as one marker or several markers. Set this value to the duration of the shortest transaction, minus several seconds.

After entering these values, click Apply Logic.

7 Verify that the markers specify the correct start times for the transactions of interest. It may be necessary to add or remove markers, or to configure the logic used for setting the markers.

If extracting transactions from multiple packet traces, each file must have the same number of markers. Otherwise, the Extraction Assistant cannot match capture data for a transaction that was captured from multiple locations.

Note—If the markers are not placed correctly, the resulting packet traces will not record the transactions accurately. For more information, see Editing Markers in the Extraction Assistant.

8 Click Extract…

➥ AppTransaction Xpert generates a set of application capture (*.appcapture) files and places them in transaction-specific subdirectories, as described in Output Files/Directories.

End of Procedure 3-3

Output Files/Directories

The Transaction Extractor creates a subdirectory in the default models directory; the subdirectory name includes the date and time of the capture-slicing operation.

For each transaction, one directory is created in the user's default model directory. Each transaction directory contains packet traces (one per capture location) that record only that transaction. Thus, the resulting model directory structure looks like this:

```
<user_default_model_directory>
      <capture_slicing_dir_with_date_and_time>
            settings.txt
            <transaction_1_dir>
                  <capture_file_1>
                  <capture_file_2>
                  . . .
            <transaction_2_dir>
                  <capture_file_1>
                    . . .
```

Editing Markers in the Extraction Assistant

Your goal in the Extraction Assistant is to ensure that every marker in the data exchange charts (upper-right pane) marks the end of the last transaction and the start of the next. The following steps outline the general workflow:

1) Click Configure Marker Logic…; then specify the source IP addresses of all hosts that initiated transactions of interest.

   The expected behavior is that, before it initiates each transaction, the host sends an ICMP ECHO ("ping") message to all hosts participating in the transaction. When you specify the addresses of the initiating hosts, AppTransaction Xpert filters out all other ping messages from other hosts.

2) In the Configure Marker Logic dialog box, enter the minimum time between tiers.

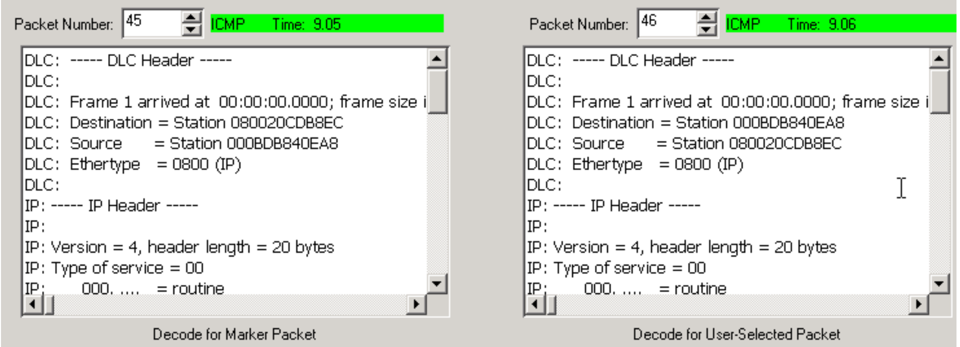
   Enter the duration of the shortest transaction, minus several seconds. The Extraction Assistant uses this value to divide the total capture window into "transaction time slices." If a time slice contains a ping message from an initiating host, AppTransaction Xpert creates a marker for that ping message. This marker represents the estimated end of the last transaction and the start of the next. (If a time slice contains multiple ping messages, AppTransaction Xpert calculates one marker based on the last ping message in the time slice.)

After you specify the source IP addresses and the minimum time between markers, click Apply Logic to return to the Extraction Assistant.

3) Edit the remaining markers and verify that they are located in the correct locations.

AppTransaction Xpert calculates marker locations based on the logic defined in the Configure Marker Logic dialog box. However, these markers represent *best estimates*. Be sure to examine each marker location and verify that it marks the actual start of the next transaction. The basic rule is:

A marker is placed correctly if and only if the packet immediately after that marker is the first packet in the transaction.



Packet Number: 45 ICMP Time: 9.05

```
DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 1 arrived at  00:00:00.0000; frame size i
DLC:  Destination = Station 080020CDB8EC
DLC:  Source      = Station 000BDB840EA8
DLC:  Ethertype   = 0800 (IP)
DLC:
IP: ----- IP Header -----
IP:
IP: Version = 4, header length = 20 bytes
IP: Type of service = 00
IP:      000. ....   = routine
```

Decode for Marker Packet

Packet Number: 46 ICMP Time: 9.06

```
DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 1 arrived at  00:00:00.0000; frame size i
DLC:  Destination = Station 000BDB840EA8
DLC:  Source      = Station 080020CDB8EC
DLC:  Ethertype   = 0800 (IP)
DLC:
IP: ----- IP Header -----
IP:
IP: Version = 4, header length = 20 bytes
IP: Type of service = 00
IP:      000. ....   = routine
```

Decode for User-Selected Packet

The marker is placed correctly…

…if the packet immediately after is the first packet in the transaction

To verify whether a marker follows this rule, select the marker; then view the contents of the marker packet in the “Decode for Marker Packet” pane (bottom-left). By default, the “Decode for User-Selected Packet” pane (bottom-right) shows the packet immediately after the marker.

You might need to do one of the following:

Move a marker—To move a marker, drag it in the exchange chart. If you know the exact packet number for a marker, select the marker and enter the packet number above the “Decode for Marker Packet” pane.

Add a marker—To create a new marker, right-click in the chart and choose Place Marker.

Delete a marker—To remove a marker, right-click on the marker and choose Delete Marker.

Related Topics

- *Capturing Application Traffic: Overview*

Capturing HTTPS Transactions for Automatic SSL Decryption

You can capture SSL-encrypted HTTP transactions and have them decrypted automatically when the packet traces are imported. For AppTransaction Xpert to decrypt these files, the following requirements must be met.

Capture Requirements—

- A license for AppTransaction Xpert Decode Module or a solution that includes this functionality.
- As of this publication, automatic decryption is supported for SSL versions 3.0 and TLS version 1.0. HTTP is the only supported protocol that runs over SSL.
- The connection cannot use SSL compression.
- The connection must use a *non-ephemeral* RSA key exchange.
- The packet trace must include the SSL handshake for the SSL-encrypted connection.

Import Requirements—

- By default, SSL uses port 443 for HTTPS. If HTTPS connections in a packet trace use a non-default port, you must edit the following file to specify the port number:

  `<install_dir>\sys\configs\adm_protocol_forces.ace_dict`

  For more information, see FAQ 1202 ("How can I force AppTransaction Xpert to decode a specific protocol on a non-standard TCP/UDP port?").
- You must copy all private keys used in encrypted transactions from the corresponding servers into one directory on the local computer.
- The key files in the specified local directory must be in PEM format and can be password-encrypted.

  For more information about converting to PEM format and additional troubleshooting steps, see FAQ 1733 ("I want AppTransaction Xpert to decrypt HTTPS transactions in my capture file, and I need to copy key files from the server to my local computer. How do I do this?).

Importing the SSL Traffic

When importing one or more packet traces with SSL-encrypted connections, you are prompted with decryption options. After clicking Next in the Merge Capture Files Dialog Box, the SSL Decryption Options dialog box appears. Choose one of the following:

- Decrypt the SSL connections and specify the local key-file directory. If any keys are password-protected, you are prompted for the password after clicking Next.
- Not to decrypt the SSL connections and continue with the import. The SSL traffic remains encrypted in the resulting Transaction Analyzer model.

Related Topics

- *Capturing Application Traffic: Overview*