SPIRENT Communications

Inspired Innovation

User Guide

Avalanche™ Attack Designer

May 2009

P/N 71-005918 REV A

Spirent Communications, Inc.
26750 Agoura Road
Calabasas, CA
91302 USA

Copyright

Limited Warranty

Spirent Communications, Inc. (“Spirent”) warrants that its Products will conform to the description on the face of order, that it will convey good title thereto, and that the Product will be delivered free from any lawful security interest or other lien or encumbrance.

Spirent further warrants to Customer that hardware which it supplies and the tangible media on which it supplies software will be free from significant defects in materials and workmanship for a period of twelve (12) months, except as otherwise noted, from the date of delivery (the “Hardware Warranty Period”), under normal use and conditions.

To the extent the Product is or contains software (“Software”), Spirent also warrants that, if properly used by Customer in accordance with the Software License Agreement, the Software which it supplies will operate in material conformity with the specifications supplied by Spirent for such Software for a period of ninety (90) days from the date of delivery (the “Software Warranty Period”). The “Product Warranty Period” shall mean the Hardware Warranty Period or the Software Warranty Period, as applicable. Spirent does not warrant that the functions contained in the Software will meet a specific requirement or that the operation will be uninterrupted or error free. Spirent shall have no warranty obligations whatsoever with respect to any Software which has been modified in any manner by Customer or any third party.

Defective Products and Software under warranty shall be, at Spirent's discretion, repaired or replaced or a credit issued to Customer's account for an amount equal to the price paid for such Product provided that: (a) such Product is returned to Spirent after first obtaining a return authorization number and shipping instructions, freight prepaid, to Spirent's location in the United States; (b) Customer provides a written explanation of the defect or Software failure claimed by Customer; and (c) the claimed defect actually exists and was not caused by neglect, accident, misuse, improper installation, improper repair, fire, flood, lightning, power surges, earthquake, or alteration. Spirent will ship repaired Products to Customer, freight prepaid, based on reasonable best efforts after the receipt of defective Products. Except as otherwise stated, any claim on account of defective materials or for any other cause whatsoever will conclusively be deemed waived by Customer unless written notice thereof is given to Spirent within the Warranty Period. Spirent reserves the right to change the warranty and service policy set forth above at any time, after reasonable notice and without liability to Customer.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, ALL IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE, ARE HEREBY EXCLUDED, AND THE LIABILITY OF SPIRENT, IF ANY, FOR DAMAGE RELATING TO ANY ALLEGEDLY DEFECTIVE PRODUCT SHALL BE LIMITED TO THE ACTUAL PRICE PAID BY THE CUSTOMER FOR SUCH PRODUCT. THE PROVISIONS SET FORTH ABOVE STATE SPIRENT'S ENTIRE RESPONSIBILITY AND CUSTOMER'S SOLE AND EXCLUSIVE REMEDY WITH RESPECT TO ANY BREACH OF ANY WARRANTY.

Contents

About this Guide

In this chapter...

Introduction

This user guide provides information on how to develop attack designs for use with the Attack List Editor in the Avalanche application.

This user guide and the Avalanche Attack Designer application are for advanced users who are experienced with network protocols and packet details. This manual assumes you are familiar with the Avalanche application.

Related Documentation

- *Avalanche application online Help*

Hardware Handling/Cleaning Practices

Electronic components are sensitive to Electrostatic Discharge (ESD) damage. To prevent premature component failure or latent product damage, it is crucial that you handle this equipment following industry standard ESD handling practices. Refer to *Appendix A, "ESD Requirements,"* for further information.

Some equipment contains fiber optic components that are very susceptible to contamination from particles of dirt and dust. Product performance may be damaged if these components are not kept clean. Refer to *Appendix B, "Fiber Optic Cleaning Guidelines,"* for proper cleaning practices for these components.

How to Contact Us

To obtain technical support for any Spirent Communications product, please contact our Support Services department using any of the following methods:

Americas

E-mail: support@spirent.com
Web: http://support.spirent.com
Toll Free: +1 800-SPIRENT (+1 800-774-7368) (US and Canada)
Phone: +1 818-676-2616
Fax: +1 818-880-9154
Hours: Monday through Friday, 05:30 to 18:00, Pacific Time

Europe, Africa, Middle East

E-mail: support@spirent.com
Web: http://support.spirent.com
Phone: +33 (0) 1 61 37 22 70
Fax: +33 (0) 1 61 37 22 51
Hours: Monday through Thursday, 09:00 to 18:00, Friday, 09:00 to 17:00, Paris Time

Asia Pacific

E-mail: supportchina@spirent.com
Web: http://support.spirent.com
Phone: 400 810 9529 (mainland China)
Phone: +86 400 810 9529 (outside China)
Fax: +86 10 8233 0022
Hours: Monday through Friday, 09:00 to 18:00, Beijing Time

The latest versions of user manuals, application notes, and software and firmware updates are available on the Spirent Communications support website at http://support.spirent.com.

Information about Spirent Communications and its products and services can be found on the main company website at http://www.spirent.com.

Company Address

Spirent Communications, Inc.
26750 Agoura Road
Calabasas, CA 91302
USA


SPIRENT
Communications

Chapter 1

Overview of Avalanche Attack Designer

This chapter provides an introduction to the Avalanche Attack Designer application.

Note: This guide and the Avalanche Attack Designer application are for advanced users very familiar with network protocols and packet details.

In this chapter...

Introduction

Avalanche Attack Designer specifies threats in an XML-based language, and the attack designs are stored as XML files. These attack design files are executed by the Attack List Editor in the Avalanche application or the Avalanche Tcl API. Attack design files can be stored and managed using the local repository, or they can be added to Spirent Communications' central database, known as the threat repository.

The Avalanche Attack Designer uses XML protocol descriptors for designing and developing attacks at the protocol level, and a PCAP dissector for importing network packet captures into the environment. See *Figure 1-1*.

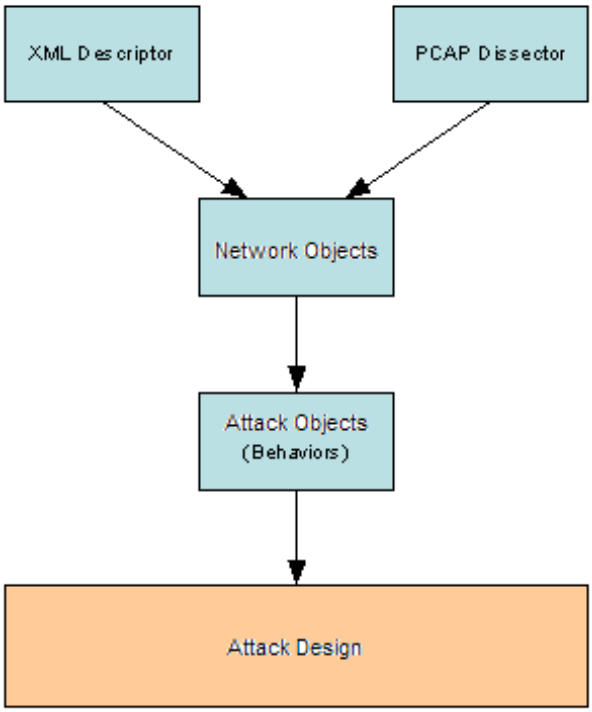


Figure 1-1. What The Avalanche Attack Designer Does

A designer first chooses a protocol-specific network object representing the target of the attack. Then, the designer enters the attack behavior for the specific network protocol. Finally, after setting the variables that may be required by the attack design, the design is saved as an XML file. The XML file is sent to the back end and used to create a stream of network packets that will simulate the attack on the device under test (DUT).

Software Licensing

You need a license file installed on the PC before you can run the Avalanche Attack Designer application. Use the License Manager Utility to generate a license request file and email it, with the sales order number, to: support@spirent.com.

To generate the Avalanche Attack Designer license request file:

1. Select **Start** > **Programs** > **Spirent** and start the License Manager Utility.
2. Select **License** > **Generate License**
   The *Generate License* option screen opens.
3. Select the **Avalanche Attack Designer** license type (radio button) and click the **Generate License Request File** button.
4. Email the imperfect.key file along with your sales order number to support@spirent.com.

To install the Avalanche Attack Designer License File:

1. Select **Start** > **Programs** > **Spirent** and start the License Manager Utility.
2. Select **License** > **Install License**.
3. Enter the full path to the license file and click **Install License File**.
4. Verify the Avalanche Attack Designer is licensed (*Figure 1-2*).

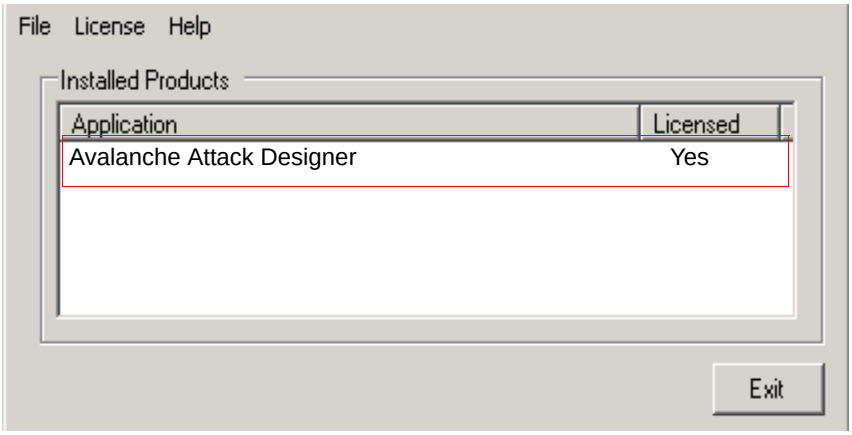


Figure 1-2. Properly Installed Avalanche Attack Designer License

Network Objects

A network object provides a virtual protocol environment from which to design any kind of attack. Each network object represents a particular network protocol, or a portion of a dynamic protocol. Within each network object is a packet description for a network protocol.

To get started, launch the Avalanche Attack Designer from your Windows® *Start* menu.

Select **Start > Programs > Spirent > Avalanche Attack Designer**.

Figure 1-3 shows some of the network objects that are available in Avalanche Attack Designer.

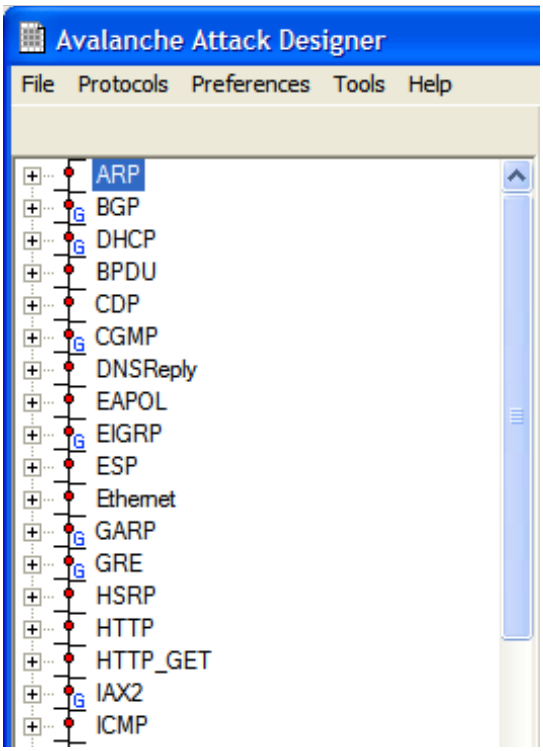


Figure 1-3. Network Objects

The Avalanche Attack Designer includes network objects for all common protocols, and new protocols are added regularly. You can also create your own protocol stack of network objects and create custom protocols, as described on *page 52* and *page 54*, respectively.

Attack Objects

An attack design is made up of a hierarchy of three basic building blocks of behavior, called attack objects. These include:

- Transmit nodes (TX) – packets of data you define to transmit as an attack
- Receive nodes (RX) – packets of data you define that are received by the attack appliance based on the attack design

Note: Attack designs that have RX nodes are known as *stateful;* attack designs that do not have any returned data are *stateless*.

- Loops – provide an attack design with repeatable behavior for TX nodes, functions, variable values, and other nested loops

You can view any existing attack design's objects that are installed with the ThreatEx option for Avalanche.

To view attack objects:

1. Select **File** > **Open** from the Avalanche Attack Designer menu.
2. Navigate to the attack repository directory.
3. Open one of the XML files that represents an attack design.
4. Select the **Attack** tab to display the attack objects, as shown in *Figure 1-4*.

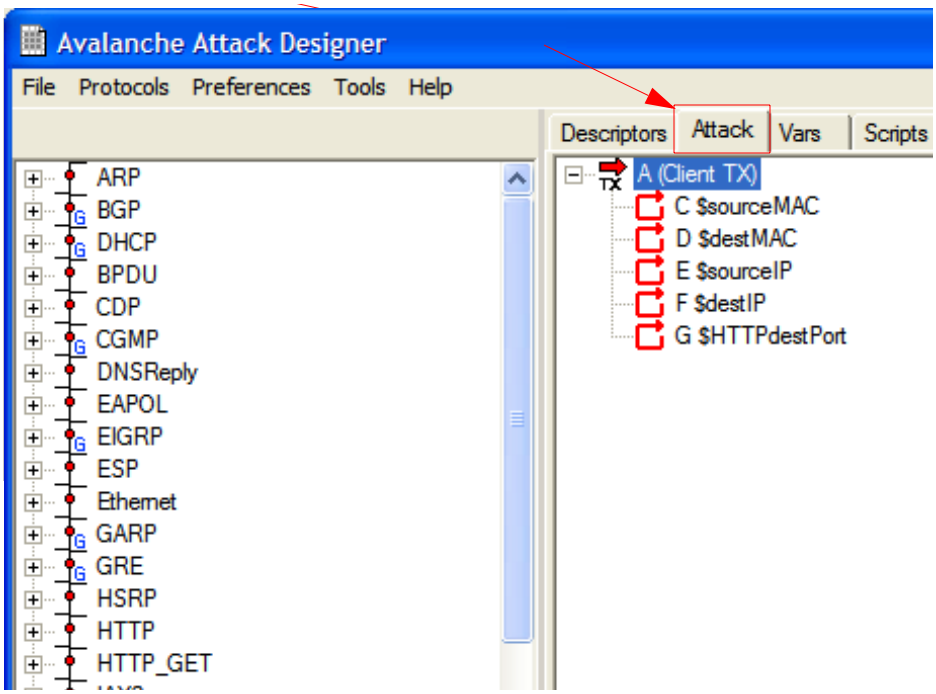


Figure 1-4. Attack Objects

Attack Object Attributes

Each attack object has attributes that are part of the protocol packet when the attack is transmitted.

To view attack object attributes:

Select an attack object from the **Attack** tab.

The attack object's attributes display on the right as a protocol stack of attributes (*Figure 1-5*.)

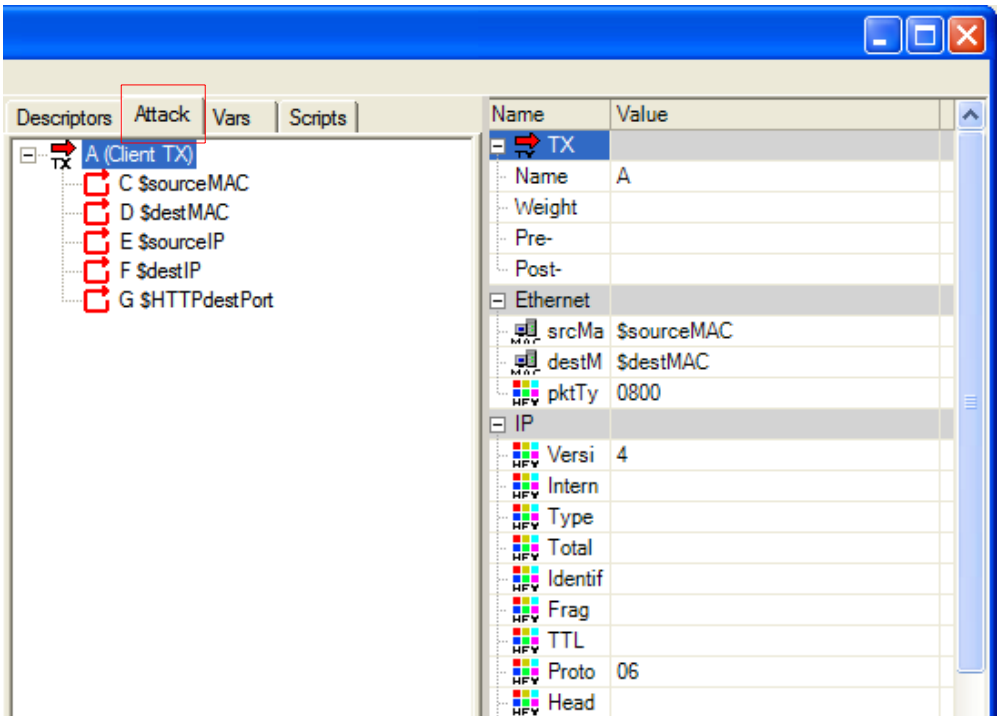


Figure 1-5. Attack Object Attributes

The basic attributes of the attack are shown in *Table 1-1 on page 15*.

Note: The *Vars* tab is discussed in *Chapter 3*.

Table 1-1. Basic Attack Object Information

Attribute	Description
Name	An identifier for the attack object.
Weight	When an attack design has more than one attack object, you can apply different weight values to define the relative number of each type of attack object generated during tests.
Pre-Script	You can specify a script to run before the attack.
Post-Script	You can specify a script to run after the attack.

The rest of the attributes each represent a field of the protocols that the attack uses. Each attribute can be assigned to a fixed value, to a function such as **@random()**, or to a variable (such as **$destIP**), the value of which is set by the user when the attack is executed.


SPIRENT
Communications

Chapter 2

Modifying an Existing Attack

This chapter provides an example of how to modify an existing attack design obtained from your local repository.

In this chapter...

Tip: Use the Attack List Editor in the Avalanche application to find existing attack designs in your local attack repository. Refer to the Avalanche online Help for more information on the local attack repository.

Modifying a Attack Design

To modify an attack design:

1 Select **File > Open** from the Avalanche Attack Designer, and select the attack design XML file you want to modify.

For this example, select the **SYNFlood.xml** attack design file.

Figure 2-1 shows the attack objects on the left and attributes of the selected attack object on the right. The SYNFlood attack has a single transmit node, named **A**. When this attack design is used in a test, the generated threats loop through a set of destination information including destination MAC addresses, IP addresses, and network ports.

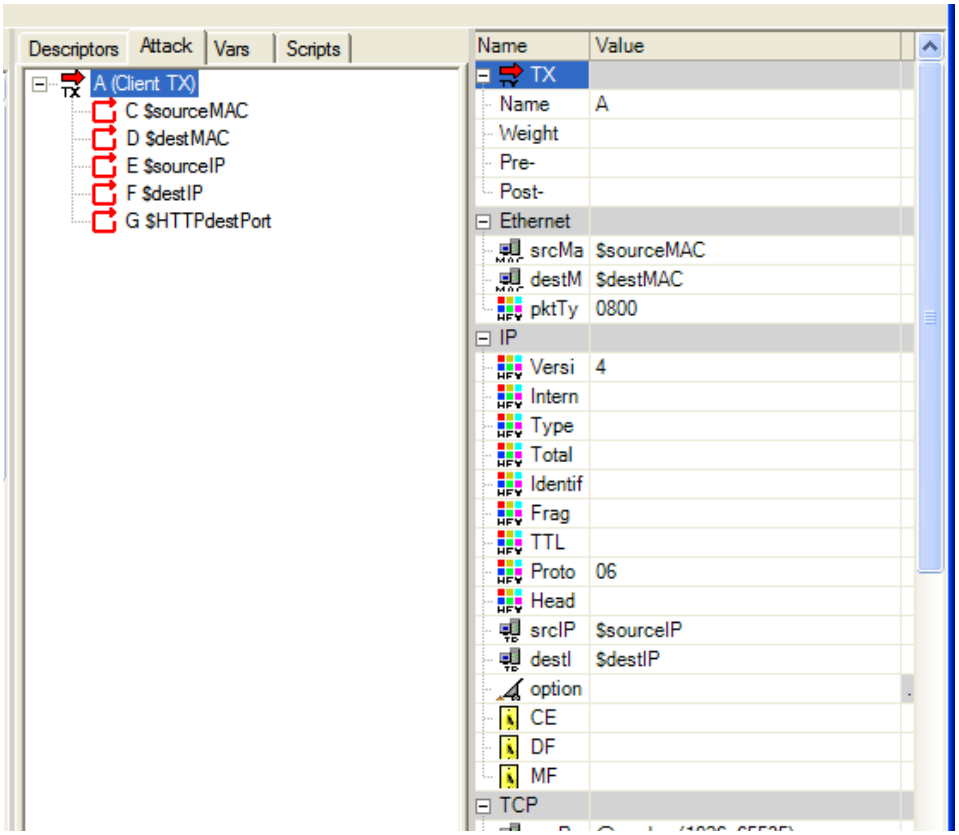


Figure 2-1. Modifying a Attack Design's Attributes

2 Modify the attack object information.

An attack object's attributes begin with basic object information described in *Table 1-1 on page 15*.

In this example, the other attributes are organized by protocol stack, beginning with the lowest Ethernet level and proceeding through the IP and TCP levels.

3 Modify the Ethernet attributes.

Change the values, as appropriate for your test, as shown in *Figure 2-2*. (In this example, no changes are necessary.)

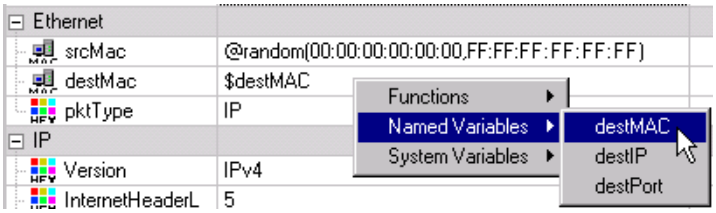


Figure 2-2. Ethernet Attributes

You can enter a value for any attribute in the protocol stack directly, select an appropriate value from a drop-down menu, or right-click an attribute to select values as follows:

- **Functions** – such as **@random()**, **@range()**, and others.
- **Named Variables** – previously defined from the *Vars* tab.
- **System Variables** – automatically defined when you add TX or RX attack objects.

4 Modify IP attributes.

Change values as appropriate for your test as shown in *Figure 2-3 on page 20*. In this example, the **srcIP** value is changed so that a subset is used instead of all possible IP values.

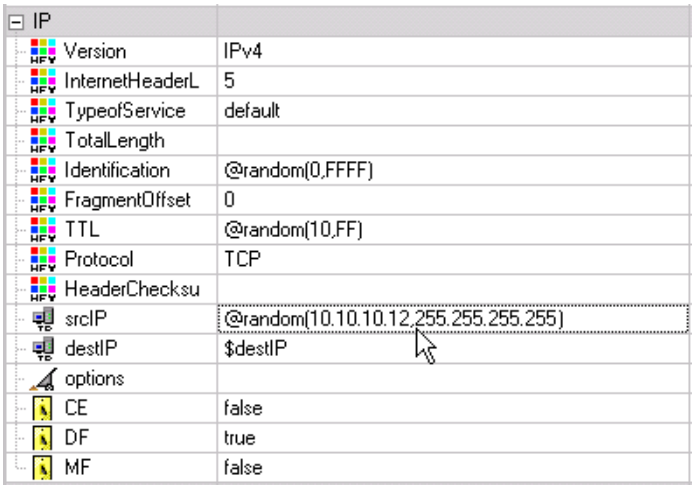


Figure 2-3. Modifying IP Attributes

5 Modify TCP attributes.

Change the values, as appropriate for your test, as shown in *Figure 2-4 on page 21*. In this example, the FIN flag value is changed to true. This change creates an attack design that behaves the same as the SYNFINflood.xml attack design, already defined for the ThreatEx option for Avalanche.

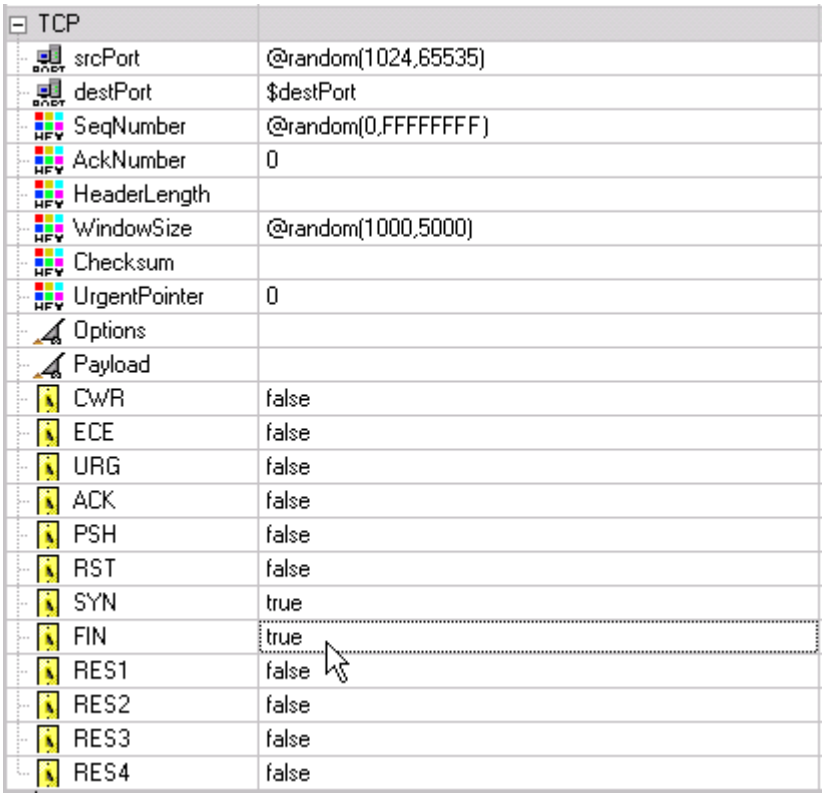


Figure 2-4. Modifying TCP Attributes

6 Modify any other protocols on the stack as appropriate.

7 Select the **Descriptors** tab and modify the design's descriptors (*Figure 2-5 on page 22*).
This step documents your changes.

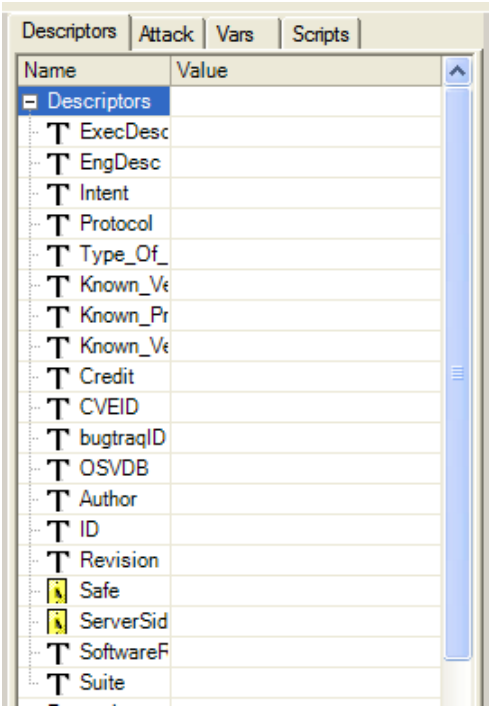


Figure 2-5. Documenting an Attack Design by Setting Descriptors

8 Select **File** > **Save**, enter a new file name, and click **Save** to preserve your design.

Using Your Modified Attack Design

Use the Avalanche Attack Designer application to load the design into a test plan and execute the attack. If the attack design will be used regularly, you can add it to your local attack repository.

Refer to the Avalanche application online Help for the procedure that describes how to add designs to your local attack repository.

Chapter 3

Creating a New Attack Design

This chapter describes how to create a new attack design from “scratch.”

In this chapter...

The tasks to create an attack design are typically performed in this order:

Selecting Attack Objects

To add an attack object design, right-click a network object. You can add a transmit (TX) or receive (RX) attack object as shown in *Figure 3-1*.

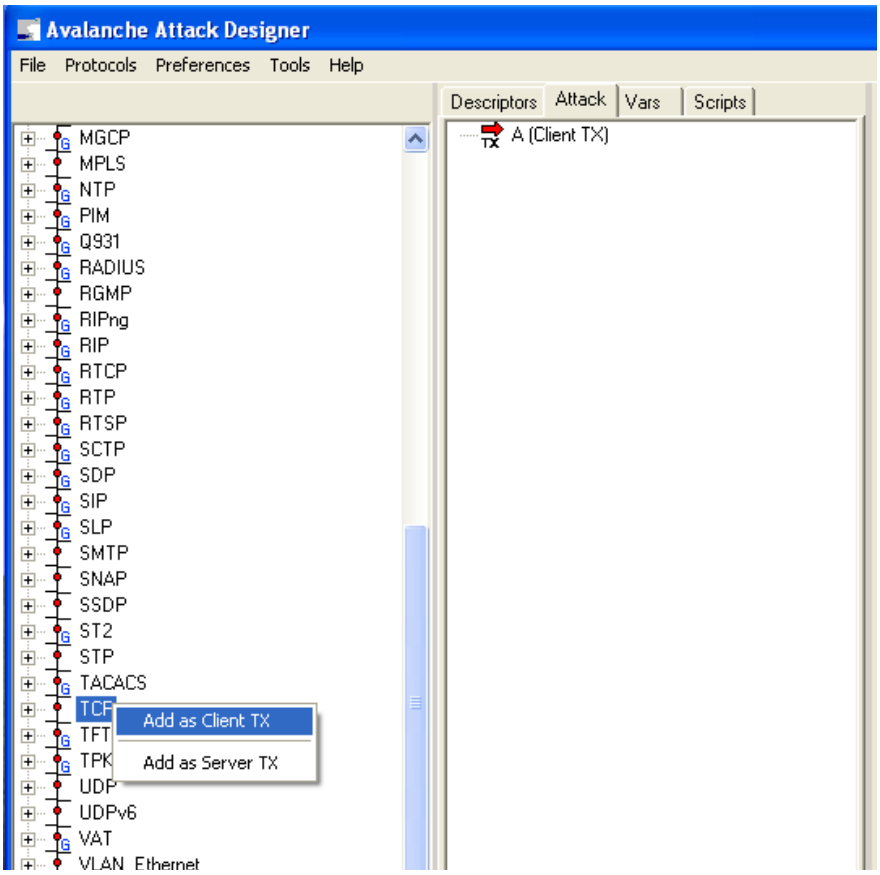


Figure 3-1. Adding a Attack Object

This example adds a single transmit TCP object. You can add different types of client and server attack objects, as described in *Table 3-1*.

Table 3-1. Types of Attack Objects

Attack Object Type	Description
Client TX	The attack object's data will be transmitted *from* clients.
Client RX	The attack object's data will be transmitted *to* clients.
Server TX	The attack object's data will be transmitted from the reflector.
Server RX	The attack object's data will be transmitted to the reflector.

Setting Loop Iteration Behavior

The hierarchy of a network object in the loop tree represents the order in which the various loops resolve. It is similar to a set of nested 'for' loops. An outer loop will increment once for an entire iteration from beginning to end of an inner loop.

To set loop behavior:

1 Right-click an attack object on the *Attack* tab to open the loop behavior context menu (*Figure 3-2*).

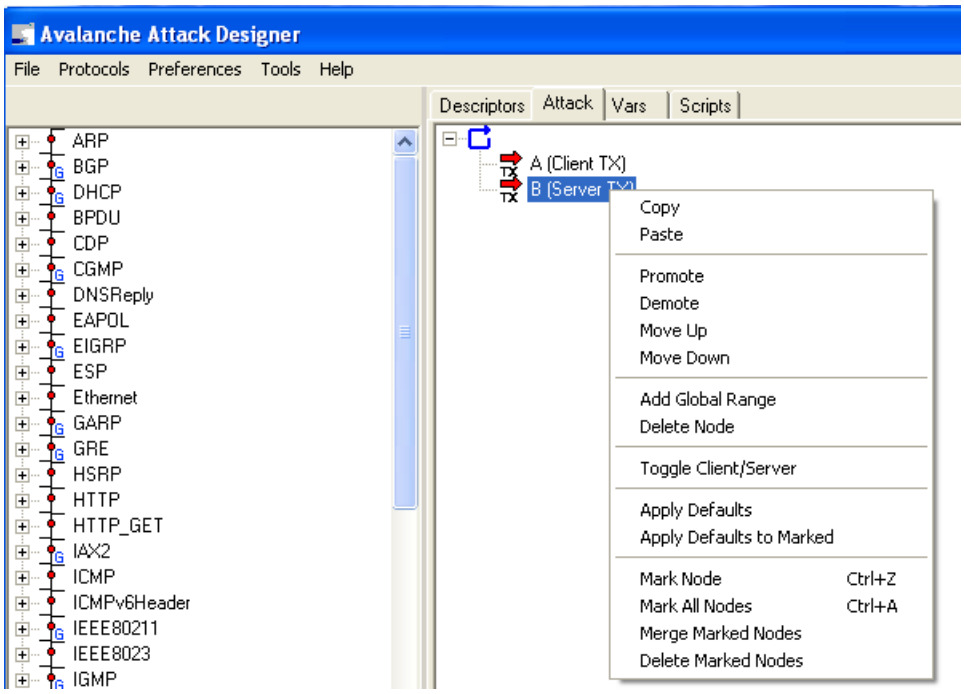


Figure 3-2. Setting Attack Object Loops

2 Adjust loop iteration behavior in any of these ways:

- Select **Promote** or **Demote** to change an object's hierarchy level.
- Select **Move Up** or **Move Down** to move the object's position within in the list of attack objects.
- Select **Add Global Range** to add a global loop in which to place one or more attack objects.
  (This example shows a global loop.)

Adding Attack Design Variables

Many attributes in the protocol stack (such as destination details) require values that are only known at runtime during an attack test run. You can create variables with Avalanche Attack Designer to use as placeholders for these types of attributes. In addition, Avalanche Attack Designer automatically defines system variables for each attack object of the attack design. You can use the system and user-defined variables for attack attribute values.

To create a variable:

1 Select the **Vars** tab (*Figure 3-3*).

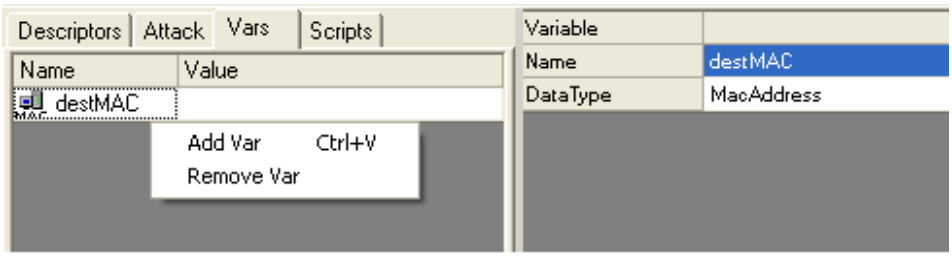


Figure 3-3. Adding Variables to a Attack Design

2 Right-click in the **Vars** tab area and select **Add Var** from the context menu.

3 Enter a **Name** for the variable in the row to the right.

4 Select a **DataType** from the drop-down list.

5 Enter a **Value** for the variable under the *Vars* tab if a default is appropriate. You can also enter a list of values.

To use a variable for an attack attribute value:

1 Select the **Attack** tab.

2 Choose a attack object and display the attack attributes to the right (*Figure 3-4*).

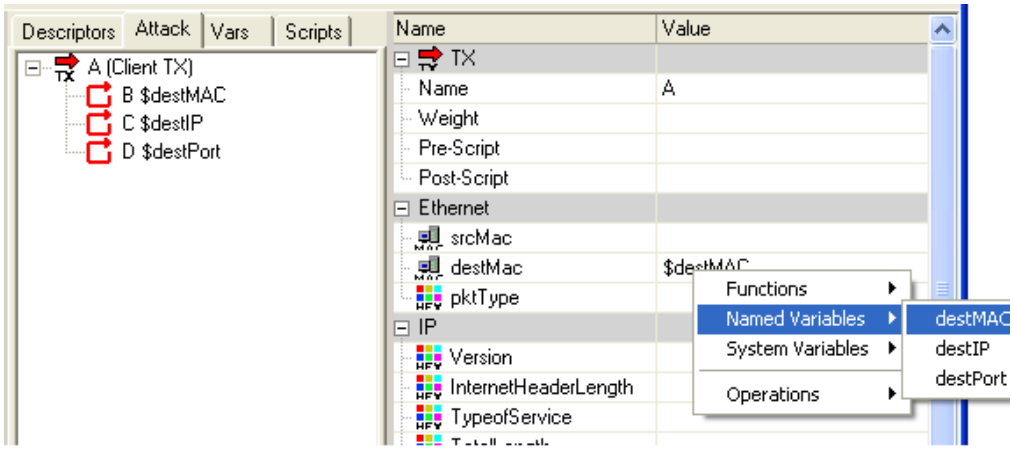


Figure 3-4. Using a Variable

3 Right-click the attribute and select the variable from the list that opens. When the variable is assigned to the attribute:

- *Named Variables* are preceded by a dollar sign (**$**)
- *System Variables* are preceded by a percent sign (**%**).

Setting Attack Attributes

The majority of attack attributes are protocol attributes organized by protocol stack beginning with the lowest Ethernet level and proceeding through the IP (or IPV6) level to the other levels such as TCP and HTTP. Set as many attribute values as are necessary to get the desired behavior. Some values have drop-down choices you can use, which are predefined in the XML.

To set attack attributes:

1. Select the **Attack** tab.
2. Select an attack object to display its attributes as a protocol stack (on the right).
3. Set basic attack object information (*Figure 3-5*).

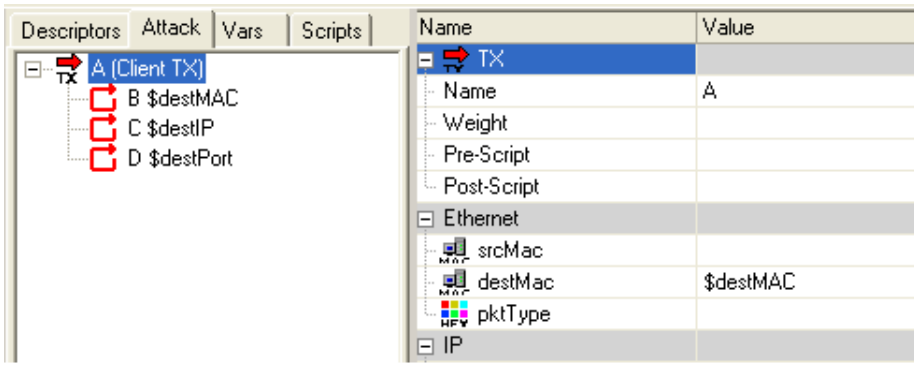


Figure 3-5. Setting Basic Attack Attributes

An attack object's attributes begin with basic object information described in *Table 1-1 on page 15*.

4. Set Ethernet values as appropriate (*Figure 3-6*.)
   Table 3-2 on page 29 describes the Ethernet attributes.

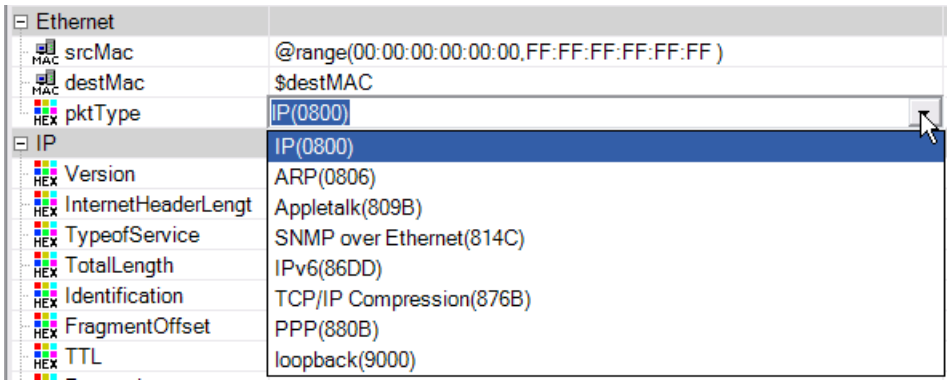


Figure 3-6. Setting Ethernet Attribute Values

Table 3-2. Ethernet Attributes

Attribute	Description
SrcMac	Typically set as random values using the **@random()** function, described on *page 33*, or set to a variable such as **$sourceMac**, so that the tester can enter the correct value at runtime.
DestMac	Typically a variable so the tester can set values for specific devices under test (DUTs). Variables are described on *page 26*.
PktType	Defines the next protocol in the stack, which is typically IP. You can select the protocol by right-clicking and selecting from the drop-down list.

5 Set IP values as appropriate (*Figure 3-7*).

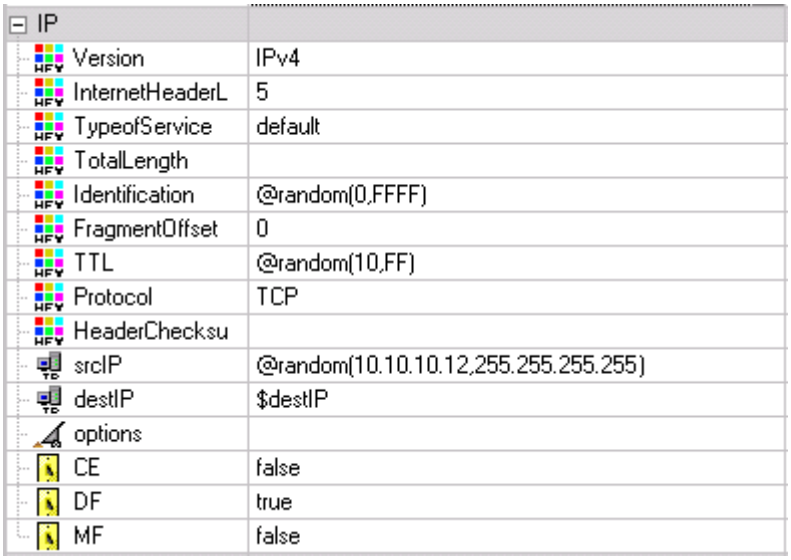


Figure 3-7. Setting IP Attribute Values

Consider the following items for IP values (*Table 3-2*):

- Set the **srcIP** value: typically set as a random value using the **@random()** function, described on *page 33*.
- Set the **destIP** value: typically a variable so the tester can set values for specific DUTs. Variables are described on *page 26*.
- Set as many other attributes as appropriate for the characteristics of the attack you are designing.

Note: This part of the protocol stack might be version IPv6, depending on the attack objects in your design.

6 Set other protocol stack values as required to achieve the desired attack behavior. Some examples of other protocols include:

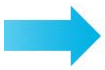
- ARP (Address Resolution Protocol)
- HTTP (Hyper Text Transfer Protocol)
- ICMP (Internet Control Message Protocol)
- IGMP (Internet Group Management Protocol)
- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)

Source attributes specify the source of the attack. They are typically generated as random values. These might include source IP, MAC, and port values.

Destination attributes specify where the attack goes. They are typically variables so the tester can set the values for specific DUTs. These might include destination IP, MAC, and port values.

Using the New Attack Design

After you create a new attack, document and save it, so you can use it in an attack test.



To use an attack design:

1 Select the **Descriptors** tab and enter information as appropriate to document your attack design.

2 Select **File** > **Save**, enter a file name, and click **Save** to save the new design.

3 Use the Attack List Editor in the Avalanche application to load the design into a test plan and execute the attack.

4 (Optional) If the attack design will be used regularly, you can add it to your local attack repository.

Refer to the Avalanche application online Help for more information on the attack repository.

Chapter 4

Using Functions for Attributes

Functions can be used to generate values at runtime for ranges, lists, and random values. Some functions, such as @fill and @randstring are designed to make it easy for the user to use the ThreatEx option for Avalanche to perform Fuzz Testing, or "fuzzing," on network protocols and services. Fuzzing is a software testing technique in which random input data is used. Fuzzing is valuable for security testing or black-box testing, and can often find oversights and defects that are commonly missed by traditional testing methods.

In this chapter...

Function Usage

To assign a function to an attack object attribute:

1 Right-click the attack object attribute field and select from the possible functions as shown in *Figure 4-1*.

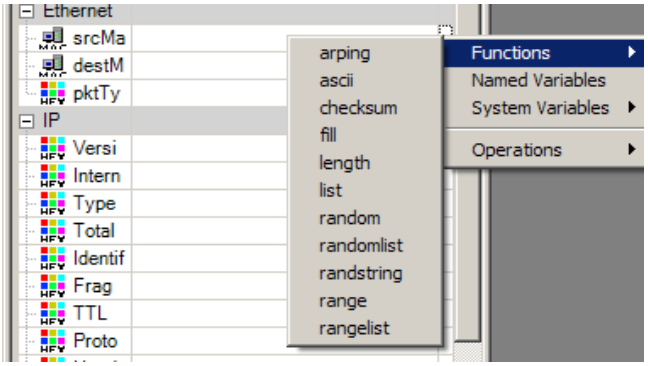


Figure 4-1. Selecting a Function

2 Enter the function parameters in the attribute's value field.

Tip: You can also type in the complete function for an attack object attribute. See *"Function Reference" on page 33* for the proper formats.

Function Reference

@random()

Description The **random()** function assigns a variable value within the specified limits in a reproducible, pseudo-random manner.

Syntax `@random(x, y)`

Comments This function can be used for IP addresses, integers, and MAC addresses. The threats generated are in a pattern that cannot be easily discerned as coming from a single source from the perspective of the networks intrusion detection and prevention system. For example, threats created with random source MAC addresses are observed by the targeted system as coming from many sources.

@range()

Description The **range()** function assigns a variable value in an iterative manner within the specified limits.

Syntax `@range(x, y)`

Comments The packets generated cover the specified bounds, looping from the minimum to the maximum value. This function can be used for IP addresses, integers, hex fields, and MAC addresses.

@fill()

Description The fill() function generates a string from a given number of repetitions of a specified string.

Syntax @fill(x, y)

Comments This function generates a new string from y concatenations of the given string "x." The second argument can be a constant or a nested range or random function.

Example @fill(ABC, 2) will generate the string "ABCABC".

Syntax @fill(x, @range(y, z))

Comments This syntax can be used to generate strings of an increasing number of repetitions of the specified string, ranging from "y" repetitions to "z" repetitions, as the attack iterates.

Example @fill(ABC, @range(1, 2)) will generate "ABC" on the first iteration and "ABCABC" on the second.

Syntax @fill(x, @random(y, z))

Comments This syntax generates a string of a random number of repetitions of the specified string, from a minimum of "y" repetitions to a maximum of "z" repetitions.

Example @fill(x, @random(1, 2)) will randomly generate either "ABC" or "ABCABC".

@randstring()

Description The randstring function generates a random string of a specified length.

Syntax @randstring(x)

Comments This function will generate a string of "x" random ASCII characters. The argument can be a constant length, or a nested range or random function.

Example @randstring(8) will generate a string of 8 random characters.

Syntax @randstring(@range(x, y))

Comments This syntax will produce random strings of an increasing length, ranging from "x" to "y".

Example @randstring(@range(1, 8)) will generate a string of 1 random character on the first iteration and 8 random characters on the last.

Syntax @randstring(@random(x, y))

Comments This syntax will produce random strings of a random length, from a minimum length of "x" to a maximum length of "y".

Example @randstring(@random(1, 8)) will generate a string of random characters with a minimum length of 1 and a maximum length of 8.

@rangelist()

Description Defines a list of elements to be iterated over as the attack runs.

Syntax @rangelist(x1, x2, ..., xN)

Comments This function can be used for any datatype, and can take any number of arguments. As the attack iterates, it will use each element in sequence from the first argument to the last.

Example @rangelist(192.168.1.1, 192.168.2.1, 192.168.4.5) will create a list of IP addresses that will be iterated over in order as the attack runs.

@randomlist()

Description Defines a set of elements to be randomly selected from.

Syntax @randomlist(x1, x2, ..., xN)

Comments This function can be used for any datatype, and can take any number of arguments. As the attack iterates, it will select and use a random element from the provided list.

Example @randomlist(25, 53, 80) could return "25", "53", or "80".

@length()

Description Calculates the length of a series of fields and constants.

Syntax @length(PROTOCOL.FIELD1, x1, ..., PROTOCOL.FIELDN [, TYPE])

Comments This function will calculate the length of the provided arguments, which can be field references, such as "IP.srcIP", or constant values. Field references will be replaced with the value of the referenced field at runtime. The optional last argument can be "BYTE", which will return length in 8-bit bytes, or "WORD", which will provide the length in 16-bit words. "BYTE" is the default type.

Example @length(IP.Options, TCP.Payload, WORD) would calculate the length of the IP Options field and the TCP Payload, measured in 2-byte words.

@checksum()

Description Calculates the checksum of a series of fields and constants.

Syntax @checksum(PROTOCOL.FIELD1, x1, ..., PROTOCOL.FIELDN [, TYPE])

Comments This function will calculate the checksum of the provided arguments, which can be field references, such as "IP.srcIP", or constant values. Field references will be replaced with the value of the referenced fields at runtime. The optional last argument can be "CRC16" or "CRC32". "CRC16" is the default type.

Example @checksum(IP.srcIP, IP.destIP, CRC16) would calculate a CRC16 checksum of the source IP and destination IP fields.

@ascii()

Description @ascii()

Syntax @ascii(arg)

Comments The payload section of an attack may require hex ASCII data to complete the required payload data. For the most part, the output format is controlled by the data type as defined in the virtual protocol for the attack protocol. However, in the case of @range and @random, the output can be ambiguous. For instance, the user may want to use @range to generate integers to be used for indexing into a random string, or to generate hex ASCII values for the payload. If the payload requires hex ASCII, the @range at @ random functions should be wrapped in an @ascii() function as follows:

Inside a hex ASCII payload
@range(1,1024) - generates integers from 1 to 1024
@ascii(@range(1,1024)) - generates hex ASCII representations of the argument (i.e. 30303031,30303032...31303234)

Chapter 5

Creating Payloads

Most protocols can be assigned a binary payload. This chapter describes how to create an encapsulated payload for an attack object attribute.

In this chapter...

Creating a Payload

To create a payload:

1 Open the Payload Editor by selecting the ellipse (**…**) from the *More* column in an attribute's row, as shown in *Figure 5-1*.

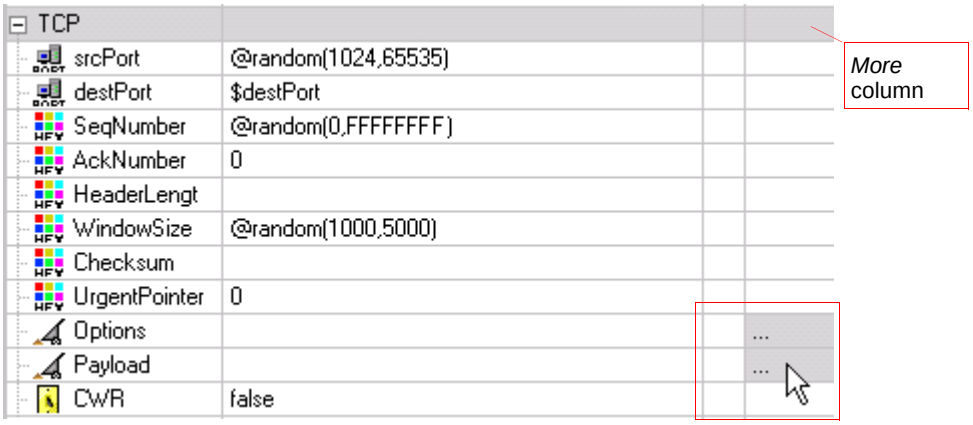


Figure 5-1. Opening the Payload Editor

The Payload Editor displays as shown in *Figure 5-2 on page 39*.

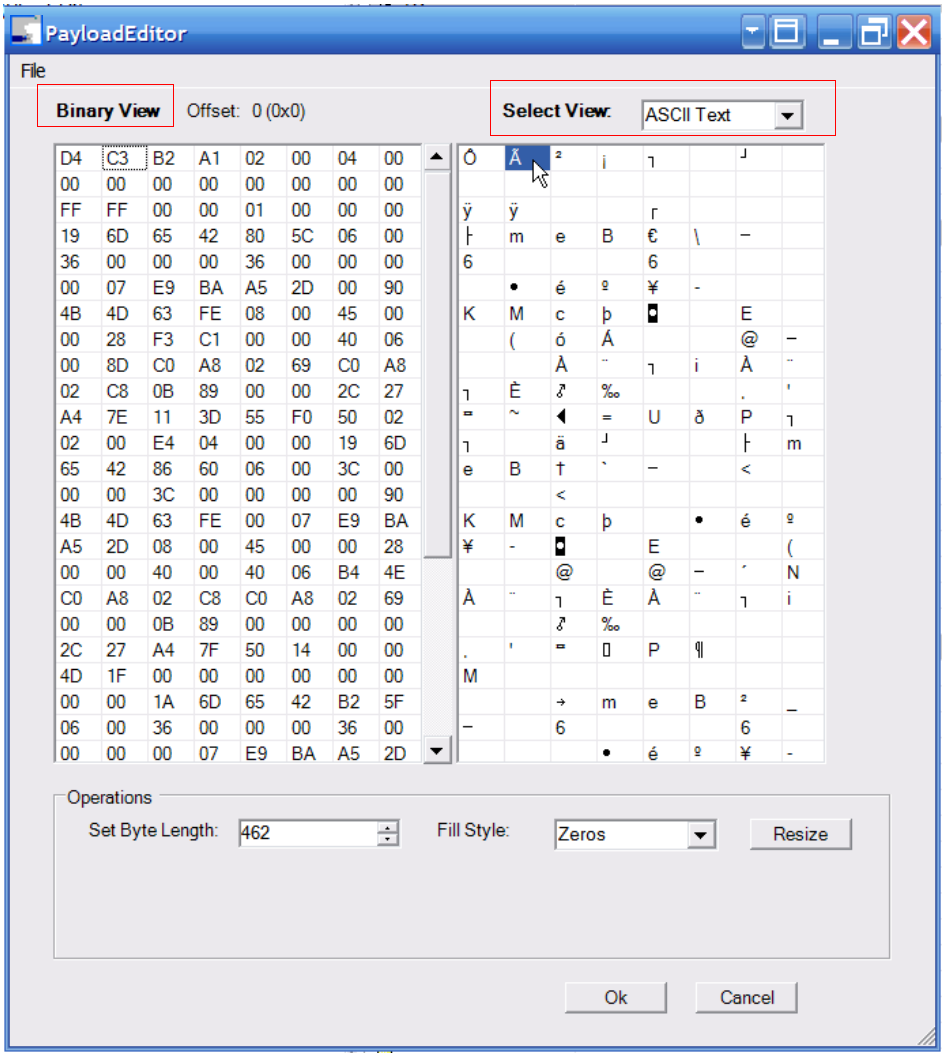


Figure 5-2. The Payload Editor

2 Select **File** > **Open** from the *Payload Editor* screen to open a binary or text file to use as a payload.

3 Use the **Select View** drop-down list to set the right-hand view to one of the following:

- ASCII Text
- Encapsulated Hex
- Unicode

You can switch between these different views as you work.

4 Click **OK** to set the payload and close the Payload Editor.

Editing a Payload

To edit a payload:

1. Select the ellipse (**…**) from the *More* column in an attribute's row to open the Payload Editor.
   The Payload Editor displays. Refer to *Figure 5-2 on page 39* as you follow the next steps.
2. To change the payload size:
   a. Enter a new payload size in the use **Set Byte Length** box.
   b. Set **Fill Style** to **Zeros** or **Random**.
   This affects the end of the payload if you increased the payload length.
   c. Click **Resize**.
3. To set specific values, select the character.
4. To insert more data from a text or binary file, use the Windows *copy* feature. For example, use this method to insert saved shell-code.
   a. Use any Windows application to select and copy the data to use (CTRL-C).
   b. In the Payload Editor, place the cursor at the point you want to insert the data.
   c. Paste the data by right-clicking and selecting **Paste** or by using CTRL-V.
5. Click **OK** to set the payload and close the Payload Editor.

Chapter 6

Importing PCAP Files

Packet capture (PCAP) files are generated by some network analyzers, including **tcpdump** and **Ethereal**. If you have a PCAP file of a known attack, you can import the PCAP file into Avalanche Attack Designer to analyze and reproduce the attack. You can then save the attack as an attack design for further tests of your equipment, or use the imported PCAP information as a basis to design other attacks.

In this chapter...

Importing a PCAP File

The following example uses a PCAP file obtained from the this Web address at Ethereal: http://wiki.ethereal.com/SampleCaptures. This file is an ARP storm PCAP that had more than 20 ARP requests per second observed on a cable modem connection.

To import a PCAP file:

1. Select **File > Import From PCAP** from the Avalanche Attack Designer.
2. Select the file **arp-storm.pcap** for this example.
   The screen shown in *Figure 6-1* displays.

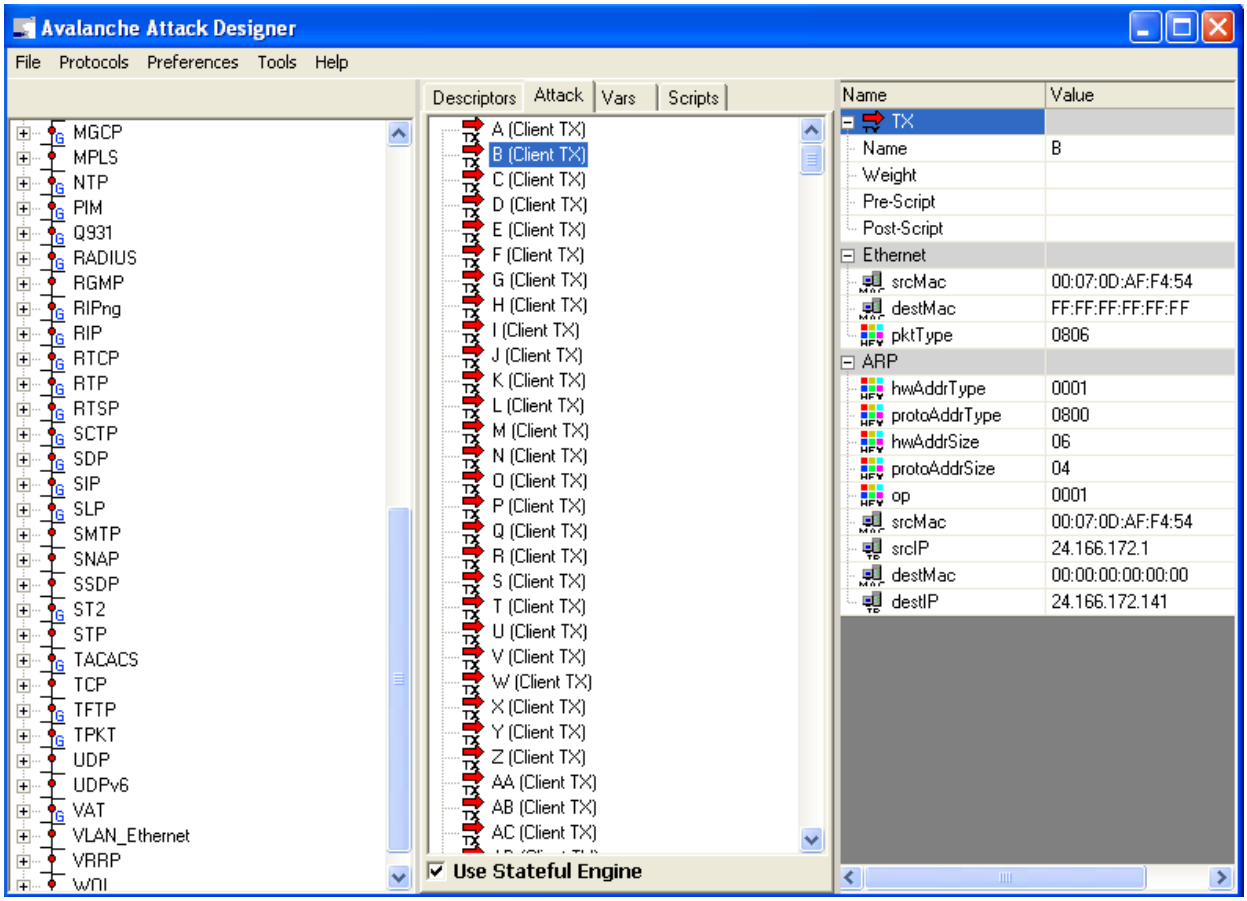


Figure 6-1. An Imported PCAP File

Importing a Multi-source PCAP File

Some PCAP files may contain captured packets originating from different sources.

To import a PCAP file with multiple sources:

1. Select **File** > **Import From PCAP** from the Avalanche Attack Designer.
2. Select the appropriate PCAP file.
   If the imported traffic represents more than one source, a window similar to *Figure 6-2* displays.

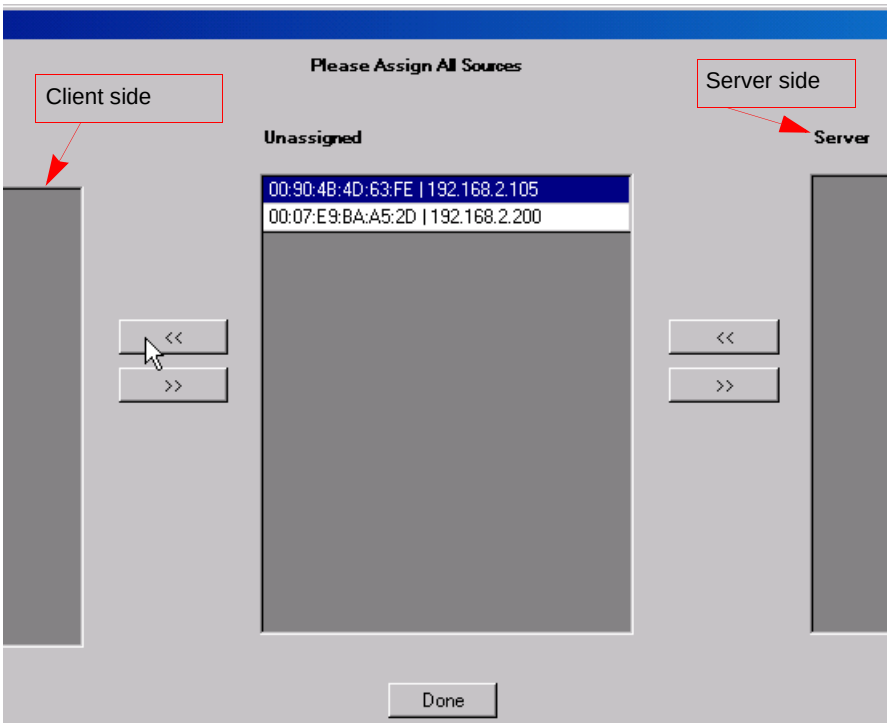


Figure 6-2. PCAP Source Assignment Window

   Avalanche Attack Designer detects the number of distinct sources represented in the capture and prompts you to assign each source to either the *client* or *server* side.
3. Use the double-arrow buttons to assign each distinct source to either the client or server (*Figure 6-2*).
   - Sources assigned to the client side will be played back from the appliance during a test.
   - Sources assigned to the server side will be played back by the reflector during a test.
4. Click **Done** to finish the assignments.

PCAP Import Settings

The Avalanche Attack Designer provides some configuration settings to automate some of the common tasks associated with creating attacks from PCAP files. These settings are divided into two sets of tasks:

- those involved with automatically ignoring fields or packets
- those involved with the automatic substitution of values or variables in imported packets.

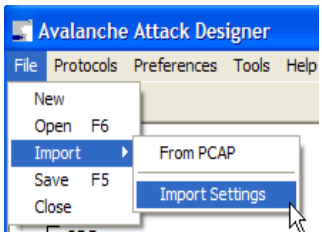


Figure 6-3. Import Settings

To configure the PCAP Import Settings:

1 Select **File** > **Import** > **Import Settings** from the Avalanche Attack Designer. The *Import Settings* screen opens (*Figure 6-4*).

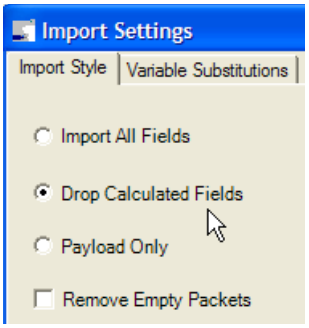


Figure 6-4. Import Styles

2 Select an import style.
The *Import Style* tab provides three import styles:

a **Import All Fields** - directs the PCAP importer to import all field values of all packets. This is the default option.

b **Drop Calculated Fields** - directs the PCAP importer to ignore the fields whose values will be automatically filled in by the stateless engine, such as the IP checksum. This style is suited for PCAP imports intended for stateless or stateful attacks.

c **Payload Only** - directs the PCAP importer to import only the payload of each packet. This style is suited for PCAP imports intended for stateful attacks.

Note: Additionally, the *Import Settings* screen allows you to enable the *Remove Empty Packets* option, which will direct the PCAP importer to automatically remove imported packets with no payload. For example, this can be a useful option if the PCAP file has a large number of empty ACK packets.

To configure Automatic Variable Substitutions:

1 Select the **Variable Substitutions** tab from the *Import Settings* screen (*Figure 6-5*).

2 Click **Add** to add a new substitution rule.

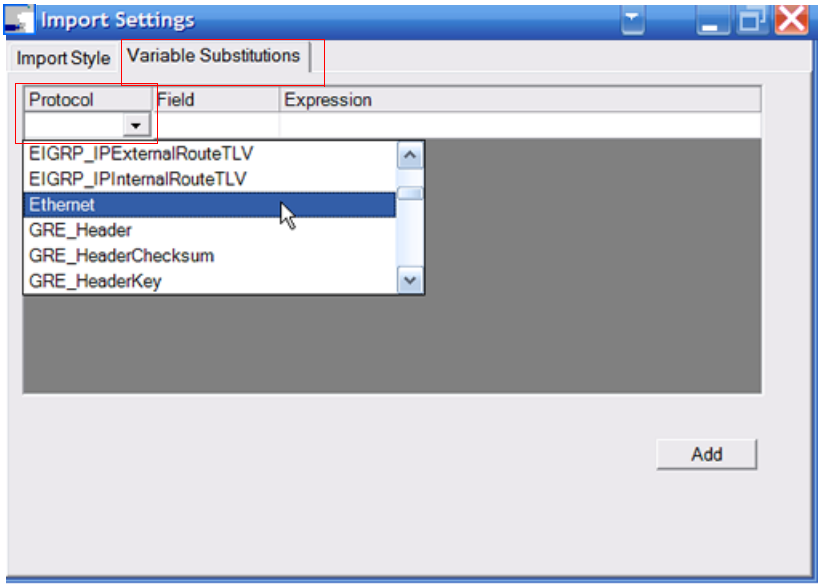


Figure 6-5. Select Target Protocol

3 Select the target **protocol** of the new rule from the drop-down menu in the *Protocol* column (*Figure 6-5*).

4 Select the target **field** of the new rule from the drop-down menu in the *Field* column (*Figure 6-6*).

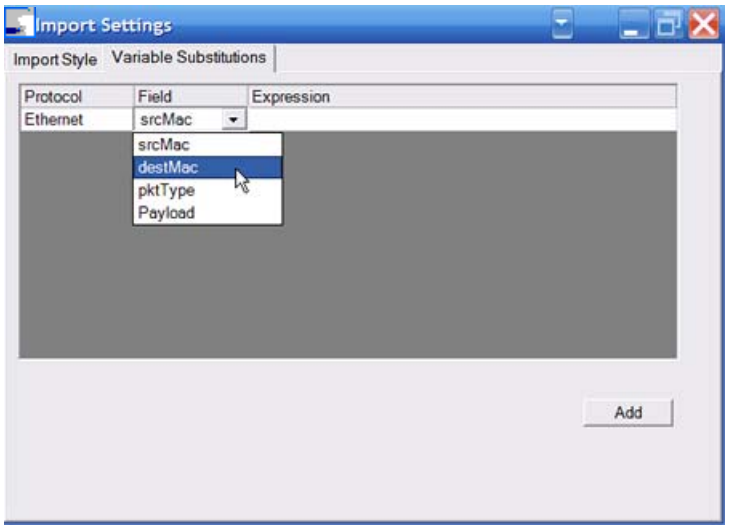


Figure 6-6. Select Target Field

5 Enter the substitution expression in the *Expression* column.

This expression will be used to replace the value of the specified field in each imported packet. The expression can be a constant value, a function, or a variable. In this example, we create a rule that will replace the *Ethernet.destMac* field of each imported packet with a variable named **destMAC**.

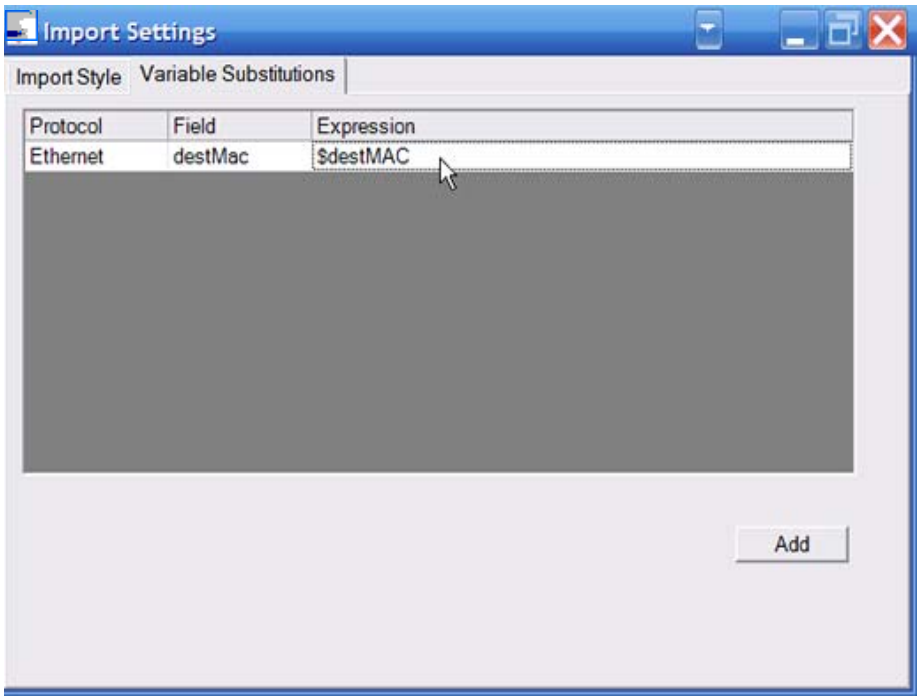


Figure 6-7. Enter an Expression

After an Import

The Avalanche Attack Designer provides several tools to aid in the creation of attacks from PCAP imports.

For situations where a series of imported nodes must be merged into one node, the Avalanche Attack Designer allows for the marking and merging of multiple nodes. Merging nodes will delete all marked nodes and replace them with a single node, whose payload will be a concatenation of the payloads of all marked nodes. For example, this is useful for manually resolving HTTP continuations.

To merge nodes:

1 Mark the nodes to be merged in the attack panel.

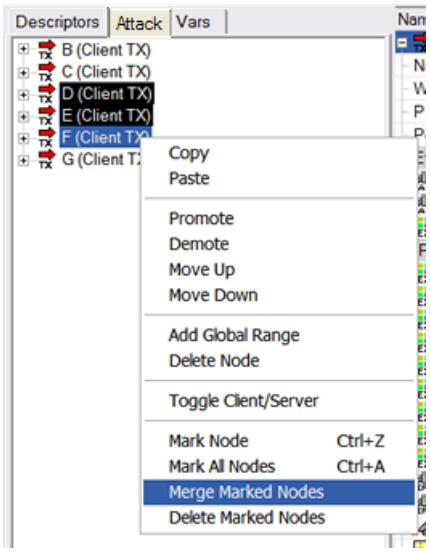


Figure 6-8. Marking and Merging Nodes

Nodes can be marked in several ways:

a Middle-click on the node

b Select the node and press **Ctrl-Z**

c Select the node and choose **Mark Node** from the context menu

d Press **Ctrl-A**, or choose **Mark All** from the context menu, to select all nodes.

2 Select **Merge Marked Nodes** after the nodes have been marked.

Collapse and Expand Operations

Additional operations can be performed on the attack attributes of attack nodes. All of the fields of a protocol can be collapsed into the payload of their encapsulating protocol, and the payload of a protocol can be expanded into a new encapsulated protocol. This section describes these procedures.

To Collapse a Protocol to a Payload:

1. Right-click in the attribute panel to open the context menu.
2. Select **Operations > Collapse to Payload**.
   This compiles all of the fields of the top-most protocol into the payload of its immediate parent.

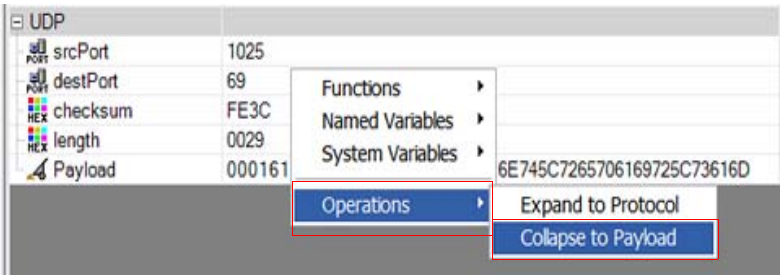


Figure 6-9. Collapsing to Payload

To Expand a Payload into a Protocol:

1. Right-click in the attribute panel to open the context menu.
2. Select **Operations > Expand to Protocol** (*Figure 6-10*).
   The *Select a Protocol* dialog opens (*Figure 6-11 on page 49*).

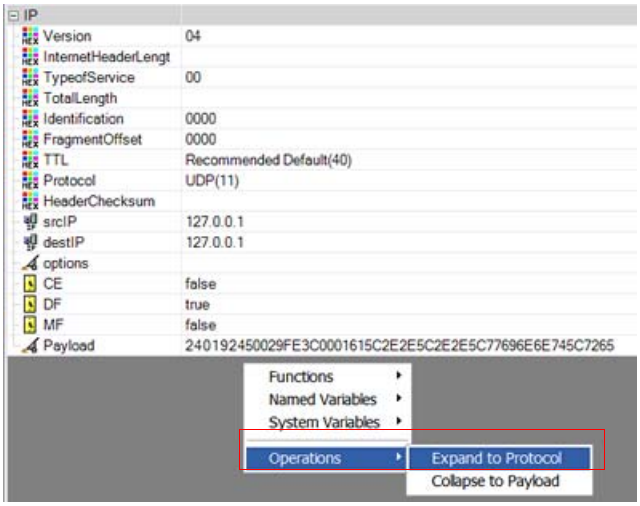


Figure 6-10. Expanding a Payload to a Protocol

3 Select the appropriate protocol from the list and click **OK**.
This creates a new encapsulated protocol object and uses the previous payload to populate its fields.

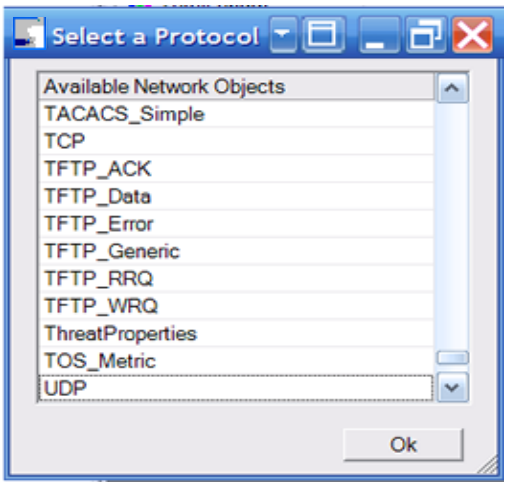


Figure 6-11. Expand To Protocol Menu


SPIRENT
Communications

Chapter 7

Creating Custom Stacks and Protocols

This chapter describes how to create your own network object stacks and protocols.

In this chapter...

Creating a Custom Network Object Stack

If the available network object stacks do not cover your situation, you can create your own. For example, you might need a custom stack for non-standard network tunneling.

To create a custom network object stack:

1 Select **Protocols** > **Create Custom Stack** to open the Custom Network Object editor as shown in *Figure 7-1*.

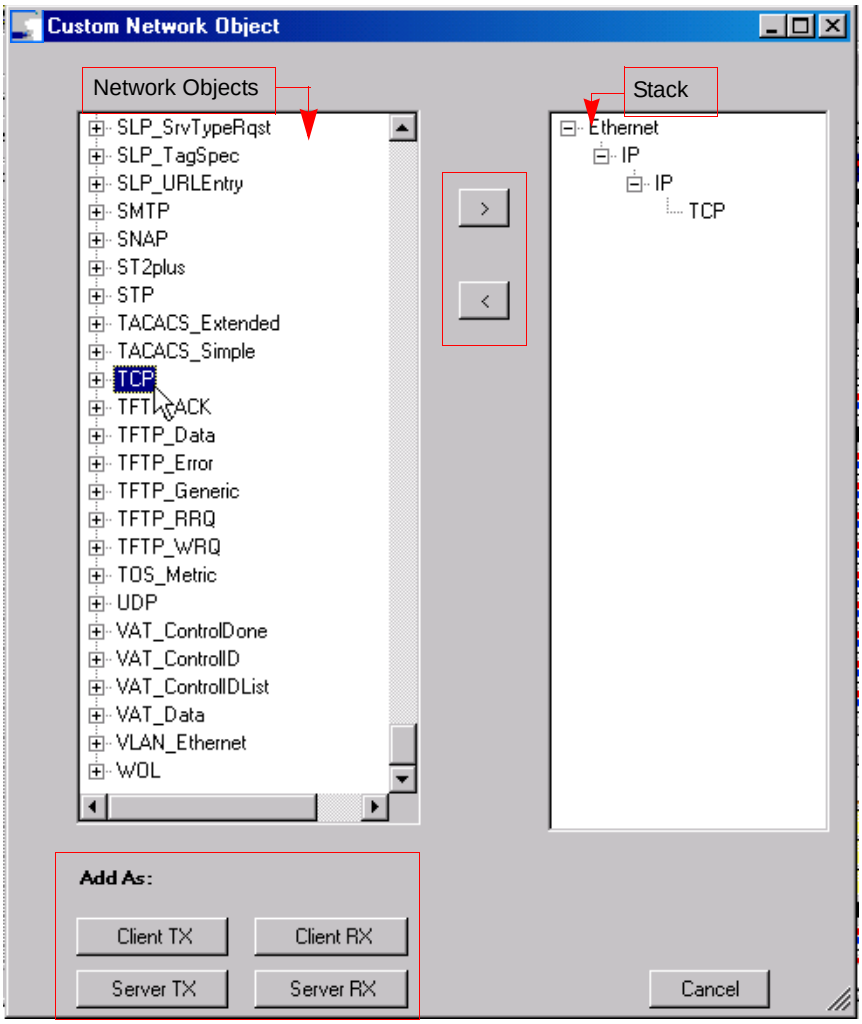


Figure 7-1. Selecting Network Objects for a Stack

2 Select a network object from the left column.

3 Click the right arrow (>) to add the network object to the stack on the right.

Start with the lowest level of your stack. (This example starts with the Ethernet network object.)

4 Repeat steps **2** and **3** for each network object.

This example demonstrates how to create a network object representing TCP traffic on top of IP-over-IP tunneling. For this example, add the network object IP, followed by IP again, and finally TCP.

5 Click one of the **Add As** buttons (*Figure 7-1 on page 52*) to finish creating the stack and close the editor.

Table 7-1 describes the *Add As* option buttons.

Table 7-1. Creating A Network Object Stack Options

Button	Description
Client TX	Adds the custom object as a client transmit node
Client RX	Adds the custom object as a client receive node
Server TX	Adds the custom object as a server transmit node
Server RX	Adds the custom object as a server receive node

Creating a Custom Protocol

If you require support for a network protocol that is not included in the ThreatEx option for Avalanche distribution, you can use Avalanche Attack Designer to add a new protocol specification. You will need this basic information about the protocol:

- An ordered list of the fields of your protocol, including their length in bits.
- The default protocol in which your protocol is usually encapsulated.
- (Optional) A list of field values that you want to appear in the drop-down menus of the Avalanche Attack Designer.

To create a protocol representation:

1 Select **Protocols** > **New Protocol** to display the New Protocol editor, as shown in *Figure 7-2*.

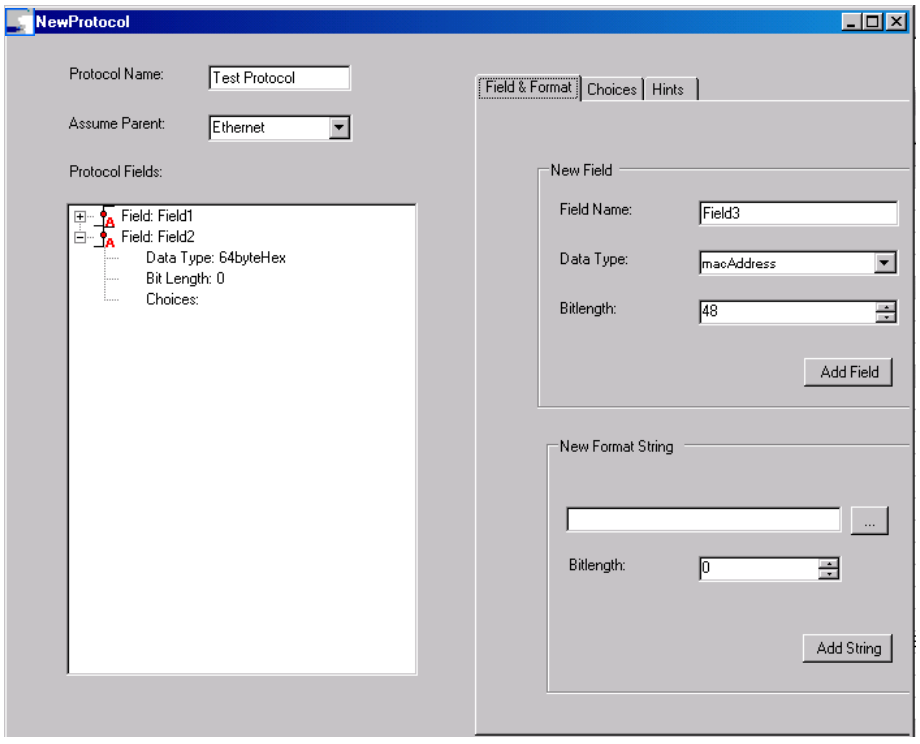


Figure 7-2. New Protocol Editor

2 Enter a **Protocol Name**.

3 Select an **Assume Parent** from the drop-down menu.
For this example use the default protocol.

4 Under the *Field & Format* tab, include the following for each field in the protocol:

   a Enter the **Field Name**.

   b Select the **Data Type** from the drop-down menu.

   c Assign a **Bitlength**.

   d Select **Add Field** to add a field to your protocol specification.

   As each new field is added, it displays in the left panel labeled *Protocol Fields*.

5 (Optional) Some protocols have their fields delimited by standard format strings. If your fields are delimited by format strings, use the *New Format String* options to insert a format string between fields.

6 (Optional) Select the **Choices** tab (*Figure 7-2 on page 54*).

   For each protocol field, you can assign some choices to appear in the drop-down menus. See *Figure 7-3*.

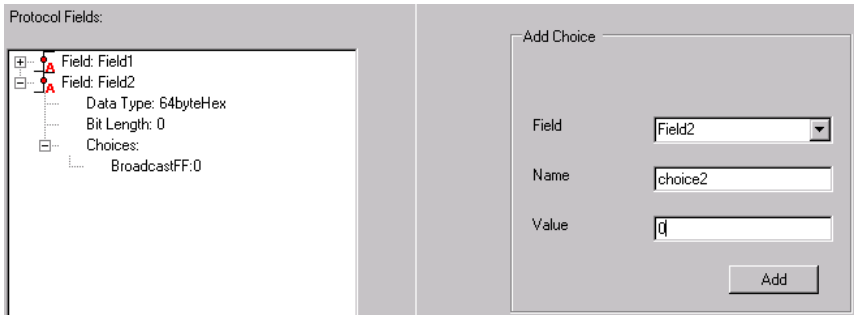


Figure 7-3. Adding Choices

   To Add Choices:

   a Choose the target field from the **Field** drop-down box.

   b Enter a Name and Value for your choice.

   The *Name* will be displayed to the user, and the *Value* will be put on the wire.

   c Click **Add** to add the choice to the field.

7 Select **Create** to generate the protocol specification when you are finished adding the fields and choices.

8 Select **Protocols > Edit User Protocols** to edit or delete a custom protocol.


SPIRENT
Communications

Appendix A

ESD Requirements

Spirent Communications manufactures and sells products that require industry standard precautions to protect against damage from electrostatic discharge (ESD). This document explains the proper process for handling and storing electrostatic discharge sensitive (ESDS) devices, assemblies, and equipment.

The requirements presented in this document comply with the EIA Standard, *ANSI/ESD S20.20-1999: Development of an Electrostatic Discharge Control Program*, and apply to anyone who handles equipment that is sensitive to electrostatic discharge. Such equipment includes, but it not limited to:

- All electronic assemblies manufactured by Spirent Communications
- Discrete and integrated circuit semiconductors
- Hybrid microcircuits
- Thin film passive devices
- Memory modules

Caution: Failure to comply with the requirements explained in this document poses risks to the performance of ESDS devices, as well as to your investment in the equipment.

General Equipment Handling

Whenever you handle a piece of ESDS equipment, you must be properly grounded to avoid harming the equipment. Also, when transporting the equipment, it must be packaged properly. Follow the requirements below to help ensure equipment protection.

- Wrist straps must be worn by any person handling the equipment to provide normal grounding.
- The use of foot straps is encouraged to supplement normal grounding. If foot straps are used exclusively, two straps (one on each foot) should be used. Note that foot straps are only applicable in environments that use ESD flooring and/or floor mats.
- Hold ESDS equipment by the edges only; do not touch the electronic components or gold connectors.
- When transporting equipment between ESD protected work areas, the equipment must be contained in ESD protective packaging. Equipment that is received in ESD protective packaging must be opened either by a person who is properly grounded or at an ESD protected workstation.

- Any racks or carts used for the temporary storage or transport of ESDS equipment must be grounded either by drag chains or through direct connection to earth ground. Loose parts that are not protected by ESD-safe packaging must not be transported on carts.

Workstation Preparation

The ideal setup for working with ESDS equipment is a workstation designed specifically for that purpose. *Figure A-1* illustrates an ESD protected workstation. Please follow the requirements listed below to prepare a proper ESD protected workstation.

- The ESD Ground must be the equipment earth ground. Equipment earth ground is the electrical ground (green) wire at the receptacles.
- An ESD protected workstation consists of a table or workbench with a static dissipative surface or mat that is connected to earth ground. A resistor in the grounding wire is optional, providing that surface resistance to ground is $\geq 10^5$ to $\leq 10^9$ Ω.
- The workstation must provide for the connection of a wrist strap. The wrist strap must contain a current limiting resistor with a value from ≥ 250K Ω to ≤ 10M Ω.
- ESD protective flooring or floor mats are required when floor-grounding devices (foot straps/footwear) are used or when it is necessary to move in between ESD protected workstations when handling ESDS equipment.

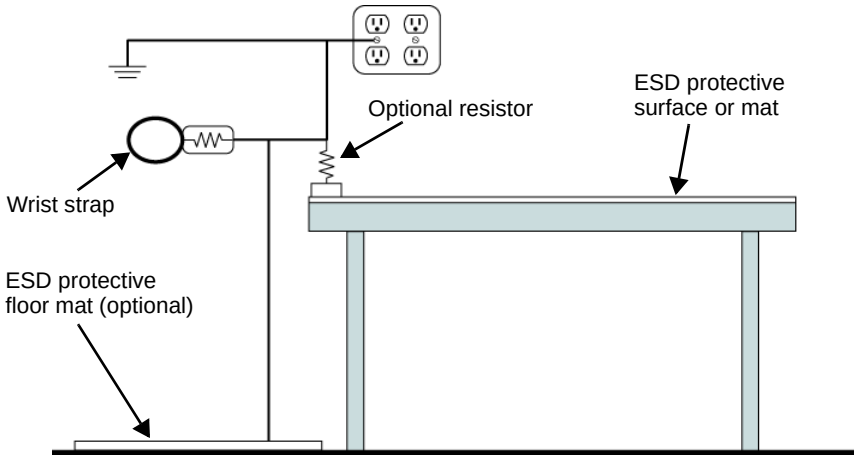


Figure A-1. ESD Protected Workstation

Note: The equipment needed for proper grounding is available in ESD service kits, such as the ESD Field Service Kit available from Spirent Communications (P/N 170-1800). Additional information on ESD can be found on the following website: http://www.esda.org/aboutesd.html

Appendix B

Fiber Optic Cleaning Guidelines

Spirent Communications manufactures and sells products that contain fiber optic components, including fiber optic transmitters and receivers. These components are extremely susceptible to contamination by particles of dirt or dust, which can obstruct the optic path and cause performance degradation. To ensure optimum product performance, it is important that all optics and connector ferrules be kept clean.

This document presents guidelines for maintaining clean fiber optic components. Spirent Communications recommends that these guidelines be followed very closely.

Caution:

- Failure to comply with the guidelines explained in this document poses risks to the performance of fiber optic-based devices, as well as to your investment in the equipment.
- Whenever you handle a piece of equipment that contains fiber optic components, you must be properly grounded to avoid harming the equipment. See the Appendix in this document titled *ESD Requirements* for more details on ESD.

Cleaning Guidelines

To ensure the cleanliness of fiber optic components, follow the guidelines below:

- Use fiber patch cords (or connectors if you terminate your own fiber) only from a reputable supplier. Low-quality components can cause many hard-to-diagnose problems during an installation.
- Dust caps are typically installed on fiber optic components to ensure factory-clean optical devices. These protective caps should not be removed until the moment of connecting the fiber cable to the device. Ensure that the fiber is properly terminated, polished, and free of any dust or dirt. Also make sure that the location of installation is as free of dust and dirt as possible.
- Should it be necessary to disconnect the fiber device, reinstall the protective dust caps.
- If you suspect that the optics have been contaminated, alternate between blasting with clean, dry, compressed air and flushing with methanol to remove particles of dirt.


SPIRENT
Communications