

Testing Specific Protocols

This document provides a PDF version of the “How to Test Specific Protocols” topics from the Avalanche Commander Help. If you want a printed version of this information, you might find it more convenient to print this document instead of the individual Help topics. For each protocol it is assumed that you:

Created a Device test if you are using Avalanche and Reflector, or client and server SmartBits modules.

Created an Application test if you are using Avalanche or a client SmartBits module with your own server or infrastructure.

Completed entries on other Commander tabs to define load, associations, and so on that are required for all Commander tests. For more information about other tabs, see Creating and Running an Advanced Test in the Help.

The following testing information is included in this document:

Testing Capture Replay	3
About Capture Replay Testing	3
Adding a Content File	3
Defining Client Capture Replay Information	4
Defining Server Capture Replay Commands	5
Adding an Actions List	6
Running the Test and Reviewing Results	7
Testing DNS	7
About DNS Testing	7
Adding an Actions List	8
Defining the Server Profile	9
Running the Test and Reviewing Results	11
Additional Examples	12
Testing FTP	16
About FTP Testing	16
Adding an Actions List	16
Defining the Server Profile	18
Running the Test and Reviewing Results	18
Testing HTTP	19
About HTTP Testing	19
Adding an Actions List	19
Defining Client Information	20
Defining the Server Profile	21
Defining Server Transaction Objects	22
Running the Test and Reviewing Results	22
Testing MM4	23
About MM4 Testing	23
Adding an Actions List	23
Syntax	23
Examples	25
Defining the Client Profile	25
Defining the Server Profile	25

Running the Test and Reviewing Results	26
Testing POP3.....	27
About POP3 Testing	27
Adding an Actions List	27
Defining the Server Profile	29
Running the Test and Reviewing Results	30
Testing RTSP/RTP Streaming	31
About RTSP/RTP Testing	31
Adding Content Files	31
Adding an Actions List	32
Defining the Server Profile	33
Running the Test and Reviewing Results	33
Testing SIP	34
About SIP Testing	34
Defining Client Information	34
Adding an Actions List	35
Adding a Content File	38
Defining the Server Profile	38
Running the Test and Reviewing Results	39
Testing SMTP	39
About SMTP Testing	40
Adding an Actions List	40
Adding Content Files	44
Defining the Server Profile	44
Running the Test and Reviewing Results	45
Testing Telnet	46
About Telnet Testing	46
Defining Client Telnet Commands.....	46
Defining Reflector Telnet Commands	47
Creating a Forms Database for a Telnet Test	49
Adding an Actions List	51
Running the Test and Reviewing Results	52
Testing VOD Multicast.....	52
About VOD Multicast Testing	52
Adding an Actions List	53
Defining the Server Profile	54
Adding Content Files	55
Running the Test and Reviewing Results	55
Testing Windows Media (MMS)	56
About Windows Media Testing	56
Adding an Actions List	56
Adding Content Files	57
Defining the Server Profile	57
Running the Test and Reviewing Results	58

Testing Capture Replay

To complete Capture Replay information and run the test:

1. Learn about Avalanche Capture Replay testing.
2. Add a content file (if you want to supply commands this way).
3. Define Capture Replay information for the client.
4. Define Capture Replay information for the server. (Device test only)
5. Add an Actions list.
6. Run the test and review results.

About Capture Replay Testing

Capture Replay testing helps you evaluate overall capacity handling, error handling, effectiveness of QoS mechanisms, and other capabilities. You can produce realistic traffic without a full implementation of the protocol by manually defining the commands to be sent, or recording a file of the protocol and replaying the communication. Avalanche supports two types of capture files:

PCAP files created using a packet capture program such as TCPDUMP or Ethereal.


CAP files created using Network Associates Sniffer 2.00x program.

NOTE: Capture Replay is a send and expect protocol that works in conjunction with Reflector (or a server). You must configure the flow of the session for both the client and the server consistently. Information sent by the client is expected by the server, and vice-versa. At the end of the session, specify close on both sides, with the value specified as **do** or **expect**. (UDP forces processing at the end of the session, so these commands can be omitted.)

Adding a Content File

If you want to supply commands by using a PCAP or CAP file, you upload the file by using the Content Files tab. The PCAP file must be libpcap-compatible, such as that generated by Ethereal or tcpdump, or CAP files created using Network Associates Sniffer 2.00x program. You can use a file to supply client and server commands.

To add content files:

1. Click the **Content Files** tab.
2. Click the **Add Content File** button . A directory dialog box appears. By default, it lists files that appear in the default Content file directory. When you first install the Avalanche software, this directory contains sample streaming files. You also store your own content files in this directory.

3. Select the file, and then click the **Add** button. The file appears in the Content Files tab.

Defining Client Capture Replay Information

Define the commands that you want to use such as Send, Expect, and Wait, or select a file that defines the commands.

To configure the client:

1. Click the **Client** tab, and then the **Profiles** tab.
2. Select the client profile with which you want to work.
3. In the Capture Replay pane, select a transport protocol and an editor type. Depending on the editor type (specify commands manually or upload capture file) different fields appear for you to define the commands.
4. Click the **New** button to create a new Capture Replay profile. (You'll reference the name of the Capture Replay profile when you define an Action list to test Capture Replay.)
5. If you select to enter commands manually, complete entries in the table that appears. For more information about commands, see Capture Replay Commands in the Help. If you select to upload a file, complete entries in the **Use Capture** fields that appear. (You will identify the Content file that you added previously.) For more information about fields, see Client Profiles Capture Replay Fields in the Help.

The following shows a profile that uses commands.

Capture Replay Client Emulation

Transport Protocol: TCP LDP ETH

Select a Capture Replay Profile: [Add] [Remove] [Refresh] [Print]

Editor Type: Specify Commands Manually Upload Capture File

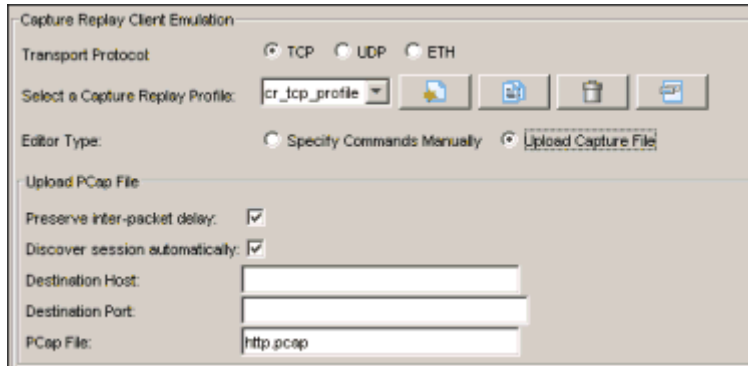
Capture Replay Commands

[New] [Copy] [Up] [Down] [Delete]

Type	Value	Min	Max
send	a	0	0
expect	b	0	0
close	expect	0	0

NOTE: To use a forms database as the source of Send commands, select the forms_db command in the Type field. Enter the column number values (1, 2, 3, and so on, without a leading \$ character) in the Value field. These values correspond to the column number in the forms database, starting from 1.


The following shows a profile that uses a file.

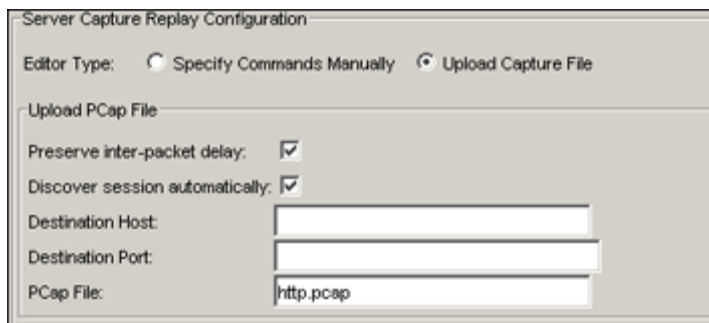


Defining Server Capture Replay Commands

Complete this section only if you are defining a Device test. You can enter server Capture Replay commands or reference a PCAP or CAP file that defines the commands. For more information about the fields, see Server Profiles Tab Capture Replay in the Help.

To configure a server profile for use with a PCAP or CAP file:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  and create a new profile.
3. Enter a description for the test, and then select **CapRepTCP** or **CapRepUDP** as the server type based on the transport type (TCP or UDP, respectively) that you configured on the Capture Replay Client Emulation pane of the Client Profiles tab.
4. Enter the port where the server resides.
5. Select the **Upload Capture File** option.
6. Select options and enter information in the Upload PCap File fields. (You will identify the Content file that you added previously.)



To configure a server profile for use with commands:

1. Perform Steps 1-4 of the previous procedure.
2. Select the **Specify Commands Manually** option. The Capture Replay Command fields appear.

3. Click the **New** button to add a new row to the table, and then enter information in the fields to define the commands.

Server Capture Replay Configuration

Editor Type: Specify Commands Manually Upload Capture File

Capture Replay Commands


New Copy Up Down Delete

Type	Value	Min	Max
expect	a	0	0
send	b	0	0
close	do	0	0

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom.

To create an Actions list:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.
3. Enter a description for the Actions list.
4. Enter information that specifies the IP address and port of the server, and the Capture Replay client profile name for TCP and UDP, respectively. See the following syntax and examples.

CAUTION: Do not include a trailing slash (/) after the IP address or an error will occur.

Syntax

```
CRTCP://ip address[:port] PROFILE=profile name [FORMS=forms database name]
```

```
CRUDP://ip address[:port] PROFILE=profile name [FORMS=forms database name]
```

IP address:port number—IP address for the test. (Port is optional; if it is not included, then 2000 is assumed.)

PROFILE=filename—Specifies the Capture Replay profile to use with this Action list. The profile name that you specify is the profile defined in the Capture Replay Client Emulation pane from the Select a Capture Replay Profile drop-down menu on the Client Profiles tab.

FORMS=filename—If you use forms database values to provide Send command values in the profile, assign the name of the database here.

Different Actions can use different forms databases, but only one forms database can be used on each separate Action.

Examples

```
CRTCP://192.168.42.11 PROFILE=capreptcp_profile
```

You can use a forms database as a source for commands, by specifying the forms_db command in the Type field on the client side in the Capture Replay Commands pane.


The following example uses a forms database:

```
CRUDP://192.168.42.11 PROFILE=caprepudp_profile  
FORMS=dbfileabc
```

Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

Testing DNS

To complete DNS-specific information and run the test:

1. Learn about Avalanche DNS testing
2. Add an Actions list
3. Define the server profile (Device test only)
4. Run the test and review results

About DNS Testing

With Avalanche, you can generate DNS queries requesting a response from either a DNS server, or Reflector or SmartBits modules emulating a DNS Server. This testing allows you to assess the capacity and functionality of DNS-aware devices, such as firewalls and DNS server infrastructures.

DNS support for Avalanche and Reflector allows

Enterprises and Service Providers to test a DNS infrastructure using Avalanche as the load generation appliance. Avalanche generates DNS queries and sends them to the DNS infrastructure.

Network Equipment Manufacturers to test the DNS correctness and capacity of their DNS-aware devices. Avalanche generates DNS traffic which travels through an intermediate device and Reflector or SmartBits act as a DNS Server responding to the queries sent by Avalanche.


Service Providers and Enterprises to evaluate network equipment during the buying process and before it is released to a production environment.

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom. Unlike Avalanche HTTP syntax, you don't specify levels for DNS because they are not relevant for DNS testing.

TIP: You can use the Create New DNS Host File window to create DNS host files that map host names to the IP addresses of URLs in an Action list.

To add an Actions list for a DNS test:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.
3. Enter information that simulates a DNS client sending different types of DNS name resolution requests to a DNS server. Refer to the following syntax and examples.

Syntax

```
DNS query_type server_IP_address name_to resolve
```

NOTE: You can also use the parameter `secondary_server_ip_address` with PTR, CNAME, MX, NS, and SOA queries and the parameters `<EXPECTS>` and `<RECURSIVE>` with all DNS query types except PTR.

Query type—The DNS query type such as A (Address) or PTR (pointer). See Defining the Server Profile later in this topic, for a complete list of query types.

Server IP address—The DNS server such as 10.1.2.3.

Secondary_server_ip_address—a secondary server.

Name to be resolved—The web site name such as `www.somewebsite.com`.

RECURSIVE—Enables the Recursive flag in the request message. It requests that the DNS server completely resolve the DNS name before

responding, instead of supplying only the first step in name resolution. The DNS server can be configured to allow or deny recursive requests. Some DNS servers are configured to deny recursive queries to avoid burdening the DNS server.

EXPECTS—Identifies the IP address expected for the DNS name specified. In the third example shown below, it expects 10.1.1.1

Secondary—a secondary DNS server that is accessed if the first url is not resolved.

Examples

Address query:

```
DNS A 10.1.2.3 www.somewebsite.com
```

```
DNS A 10.1.2.3 www.somewebsite.com <RECURSIVE>
```

```
DNS A 10.1.2.3 www.somewebsite.com <EXPECTS 10.1.1.1>
```

Optional secondary DNS support feature:

```
DNS A 192.168.42.11 somewebsite.com 192.168.42.12
```

This results in accessing 192.168.42.11 and attempting to resolve somewebsite.com. If this fails, 192.168.42.12 (secondary DNS server) will be accessed, and another attempt made to resolve somewebsite.com.


TIP: See Additional Examples, later in this topic for other example syntax and explanation.

NOTE: Direct access of a forms database is not supported by the DNS structure. However, you can use the **ASSIGN** variable and forms database to build requests with a DNS protocol. For more information, see Using Form Databases with DNS, later in this topic.

Defining the Server Profile

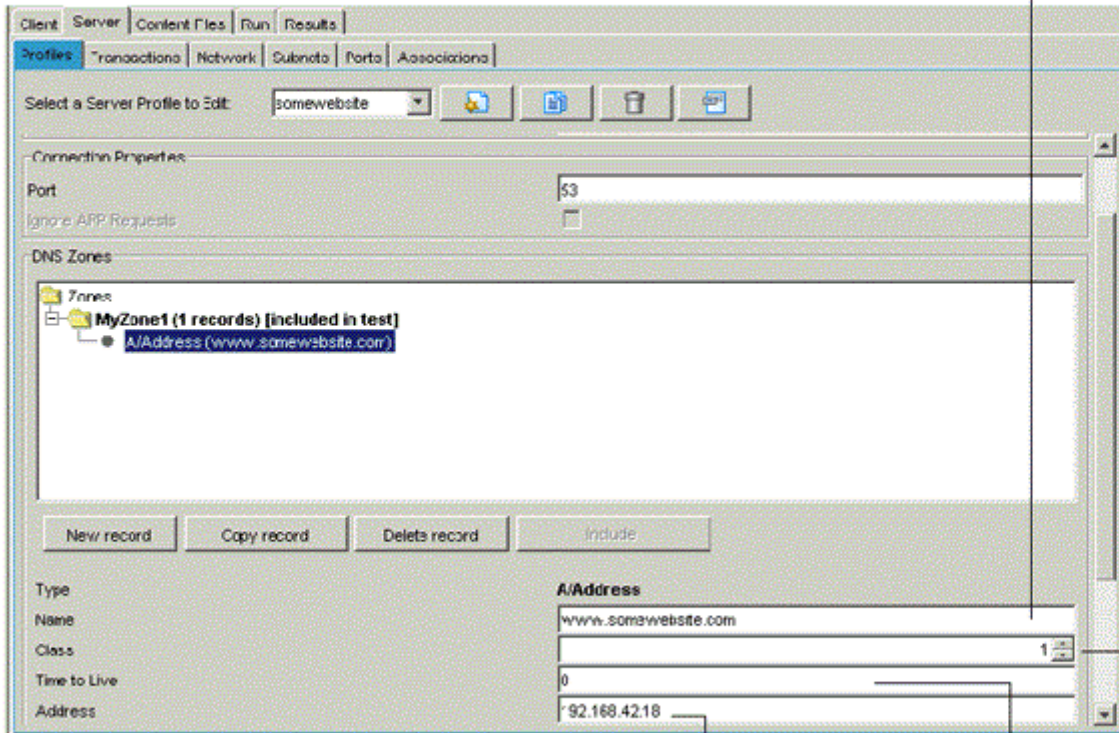
Complete this section only if you are defining a Device test.

To define a DNS server profile:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
4. Enter a description for the test, and then select **DNS** as the server type.
5. Enter the port where the server resides. The default port is 53.
6. Click the **New Zone** button. Zones are the equivalent of DNS zones. They help you organize the different DNS queries that you create. (If you have more than one zone, you must close an open zone before you can add a new one.)
7. Click the **New Record** button, and then select a record type (DNS query) from the drop-down list that appears. Avalanche supports the following types of DNS queries:

- Address (A). Maps domain names to IPv4 addresses.
 - Address (AAAA). Maps domain names to IPv6 addresses.
 - Canonical Name (CNAME). DNS equivalent of an alias or symbolic link.
 - Mail Exchange (MX). Marks a server that has been designated as the mail server for a specific domain.
 - Name Server (NS). Marks the beginning of a DNS zone and supplies the domain name of a name server for that zone. Typically you see it at the top of a zone, just after the SOA, and at the start of a subzone, where an NS and a paired A are all that is required to perform delegation.
 - Pointer (PTR). Similar to CNAME in format, however, CNAME specifies an alias, while a PTR points to another location in the domain name space. The most important use of PTRs is to construct the in-addr.arpa domain, used to convert IP addresses to DNS names (the reverse of the normal process).
 - Start of Authority (SOA). Marks the beginning of a DNS zone, and is typically seen as the first record in a name server for that domain.
8. Complete entries in the fields that appear to support your record type. For definitions of all fields, see the Server Profiles Tab DNS Help. The following shows an example of an address query.

1. Enter the name of the host that owns the resource record (data within the DNS zone). Reflector compares the Name string that you specify with that in the Actions list, and responds only if there is an exact match.



2. Select the Class. DNS supports only the 1 (IN Internet) class of record.


4. Enter the IP address to be mapped.

3. Enter the amount of time (in seconds) the resource record can be stored in cache.

Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

NOTE: In the resulting statistics, each attempt is considered as attempted, and successful or unsuccessful. If the attempt was resolved on the first DNS server, the result is one attempted and one successful request. If the attempt was not resolved on the first DNS server, but resolved on the second DNS server, the result is two attempted, one successful, and one unsuccessful request. If both attempts fail, the result is two attempted and two unsuccessful requests.

Additional Examples

The following provides additional Action List examples for testing against a DNS server and for testing between Avalanche and Reflector.

Testing Against a DNS Server

The following provides examples for testing against a DNS server.

A Record

Contacts the DNS server 192.168.42.11 to resolve the IP address for www.somewebsite.com. If the DNS server fails to respond, or no A record exists for www.somewebsite.com, the transaction fails.

```
DNS A 192.168.42.11 www.somewebsite.com
```

Contacts the DNS server 192.168.42.11 to resolve the IP address for www.somewebsite.com, requesting the DNS server to completely resolve the IP address for www.somewebsite.com. If the DNS server fails to respond, no A record exists for www.somewebsite.com, or the DNS server refuses recursive requests, the transaction fails. Note that recursion is disabled on some DNS servers, so this test works only when the DNS server supports recursive requests.

```
DNS A 192.168.42.11 www.somewebsite.com <RECURSIVE>
```

Contacts the DNS server 192.168.42.11 to resolve the IP address for www.somewebsite.com, expecting DNS to return the IP address 192.168.100.10 as the address for www.somewebsite.com. If the DNS Server fails to respond, no A record exists for www.somewebsite.com or any other address is returned, the transaction fails.

```
DNS A 192.168.42.11 www.somewebsite.com <EXPECTS  
192.168.100.10>
```

Contacts the DNS server 192.168.42.11 to resolve the IP address for www.somewebsite.com. If the DNS server fails to respond or no A record exists for www.somewebsite.com on the first DNS server, a second request is made to DNS server 192.168.42.12. If the secondary DNS server fails to respond or no A record exists for www.somewebsite.com on the second DNS server, the transaction fails.

```
DNS A 192.168.42.11 www.somewebsite.com 192.168.42.12
```

AAAA Record

When using the AAAA query, an IPv6 address is expected for the DNS name. The following example contacts the DNS server 10.1.2.3 to resolve the IP address for www.somewebsite.com, expecting DNS to return the IP address 2108::0200:FF:FE00:8301 as the address for www.somewebsite.com. If the DNS Server fails to respond, no AAAA record exists for www.somewebsite.com or any other address is returned, the transaction fails.

```
DNS AAAA 10.1.2.3 www.somewebsite.com <EXPECTS  
2108::0200:FF:FE00:8301>
```

PTR Record

Contacts the DNS server 192.168.42.11 to resolve the DNS host name belonging to the IP address 192.168.100.10. If the DNS server fails to respond, or no PTR record exists for 192.168.100.10, the transaction fails.

```
DNS PTR 192.168.42.11 192.168.100.10
```

CNAME Record

Contacts the DNS server 192.168.42.11 to resolve any alternative host name(s) for server1.somewebsite.com. If the DNS server fails to respond, or no CNAME record(s) exists for server1.somewebsite.com, the transaction fails.

```
DNS CNAME 192.168.42.11 server1.somewebsite.com
```

MX Record

Contacts the DNS server 192.168.42.11 to resolve the IP address for the mail server at somewebsite.com. If the DNS server fails to respond, or no MX address entry exists for somewebsite.com, the transaction fails.

```
DNS MX 192.168.42.11 somewebsite.com
```

NS Record

Contacts the DNS server 192.168.42.11 to resolve the IP address for the name server at somewebsite.com. If the DNS server fails to respond, or no NS address entry exists for somewebsite.com, the transaction fails.

```
DNS NS 192.168.42.11 somewebsite.com
```

SOA Record

Contacts the DNS server 192.168.42.11 to resolve the IP address for the DNS server which functions as the Start Of Authority for somewebsite.com. If the DNS server fails to respond, or no SOA address entry exists for somewebsite.com, the transaction fails.

```
DNS SOA 192.168.42.11 somewebsite.com
```

Testing Between Avalanche and Reflector

The following commands show example syntax for testing between Avalanche and Reflector. Avalanche and Reflector perform a literal request and response using the DNS protocol. DNS testing verifies if the devices between Avalanche and Reflector are capable of passing or blocking DNS requests.

A Record

Contacts the Reflector DNS server 192.168.42.11 to resolve the IP address for www.somewebsite.com. If the Reflector DNS server fails to respond, or no A record exists on the Reflector DNS server for www.somewebsite.com, the transaction fails.

```
DNS A 192.168.42.11 www.somewebsite.com
```

Contacts the Reflector DNS server 192.168.42.11 to resolve the IP address for www.somewebsite.com, requesting the DNS server to completely resolve the IP address for www.somewebsite.com. If the

Reflector DNS server fails to respond, or no A record exists on the Reflector DNS server for `www.somewebsite.com` or on Reflector, the transaction fails. Note that recursion is irrelevant to testing between Avalanche and Reflector, because Reflector always returns the full IP address.

```
DNS A 192.168.42.11 www.somewebsite.com <RECURSIVE>
```

Contacts the Reflector DNS server `192.168.42.11` to resolve the IP address for `www.somewebsite.com`, expecting the Reflector DNS server to return the IP address `192.168.100.10` as the address for `www.somewebsite.com`. If the Reflector DNS Server fails to respond, no A record exists on the Reflector DNS server for `www.somewebsite.com`, or any other address is returned, the transaction fails.

```
DNS A 192.168.42.11 www.somewebsite.com <EXPECTS
192.168.100.10>
```

Contacts the Reflector DNS server `192.168.42.11` to resolve the IP address for `www.somewebsite.com`. If the Reflector DNS server fails to respond or no A record exists on the Reflector DNS server for `www.somewebsite.com` on the first DNS server, a second request is made to Reflector DNS server `192.168.42.12`. If the secondary Reflector DNS server fails to respond or no A record exists on the Reflector DNS server for `www.somewebsite.com` on the second DNS server, the transaction fails.

```
DNS A 192.168.42.11 www.somewebsite.com 192.168.42.12
```

PTR Record

Contacts the Reflector DNS server `192.168.42.11` to resolve the DNS host name belonging to the IP address `192.168.100.10`. If the Reflector DNS server fails to respond, or no PTR record exists on the Reflector DNS server for `192.168.100.10`, the transaction fails. Note that because Reflector echoes the request from the server, the `in-addr.arpa` address format must be used for the transaction, both on Avalanche and Reflector.

```
DNS PTR 192.168.42.11 10.100.168.192.in-addr.arpa
```

CNAME Record

Contacts the Reflector DNS server `192.168.42.11` to resolve any alternative host name(s) for `server1.somewebsite.com`. If the Reflector DNS server fails to respond, or no CNAME record(s) exists on the Reflector DNS server for `server1.somewebsite.com`, the transaction fails.

```
DNS CNAME 192.168.42.11 server1.somewebsite.com
```

MX Record

Contacts the Reflector DNS server `192.168.42.11` to resolve the IP address for the mail server at `somewebsite.com`. If the Reflector DNS server fails to respond, or no MX address entry exists on the Reflector DNS server for `somewebsite.com`, the transaction fails.

```
DNS MX 192.168.42.11 somewebsite.com
```

NS Record

Contacts the Reflector DNS server 192.168.42.11 to resolve the IP address for the name server at somewebsite.com. If the Reflector DNS server fails to respond, or no NS address entry exists on the Reflector DNS server for somewebsite.com, the transaction fails.

```
DNS NS 192.168.42.11 somewebsite.com
```

SOA Record

Contacts the Reflector DNS server 192.168.42.11 to resolve the IP address for the Reflector DNS server which functions as the Start Of Authority for somewebsite.com. If the Reflector DNS server fails to respond, or no SOA address entry exists on the Reflector DNS server for somewebsite.com, the transaction fails.

```
DNS SOA 192.168.42.11 somewebsite.com
```

Using Forms Databases with DNS

Direct access of a forms database is not supported by the DNS structure. You can, however, use the ASSIGN variable and forms database to build requests with a DNS protocol. The following provides an example of assigning a forms database:

```
ASSIGN VARIABLE <Rec dnsdb.$1>  
ASSIGN VARIABLE <Lookup dnsdb.$2>  
ASSIGN VARIABLE <Expect dnsdb.$3>
```

You can then use the variables in an Action list by using the APPLY action:

```
DNS <APPLY Rec> 192.168.44.11 <APPLY Lookup> <EXPECT <APPLY  
Expect>>
```

An example forms database supporting the previous Action list is as follows:

```
A, www.somewebsite1.com,1.1.1.1  
A, www.somewebsite3.com,3.3.3.3  
A, www.somewebsite4.com,4.4.4.4  
A, www.somewebsite5.com,5.5.5.5  
A, www.somewebsite6.com,6.6.6.6  
A, www.somewebsite7.com,7.7.7.7  
A, www.somewebsite8.com,8.8.8.8  
A, www.somewebsite9.com,9.9.9.9  
A, www.somewebsite10.com,10.10.10.10
```

Testing FTP

To complete FTP-specific information and run the test:

1. Learn about Avalanche FTP testing.
2. Add an Actions list.
3. Define the server profile. (Device test only)
4. Run the test and review results.

About FTP Testing


Avalanche uses FTP to perform a simple file transfer from a specified server. With Reflector or SmartBits, you can emulate an FTP server, establishing a connection with a client, returning various files in binary mode, and then terminating the session. Only passive FTP is supported. To test FTP, an FTP Action list uses two elements: Level and URI. You can also use APPLY with an FTP action list to apply a previously defined variable value.

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom. You can create an Action list for Application tests or for Device tests.

For each FTP Action list entry, Avalanche establishes a connection, requests a file transfer, and then terminates the connection when it has completed. Avalanche supports only passive FTP.

To create an Actions list for an FTP application test (client emulation only):

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.
3. Enter a description for the Actions list.
4. Enter information that simulates an FTP transfer. For each FTP entry, enter 1 preceding the URI (to signify a top-level retrieval) and then the action. Refer to the following syntax and examples.

Syntax

```
1 ftp://server ip address /filename <USER=username  
PASSWD=password> <MODE=binary>
```

or

```
1 ftp://server ip_address /filename <USER_VAR=<APPLY  
user_variable> PASSWD_VAR=<APPLY password_variable>>  
<MODE=binary>
```

Server IP address—The address of the server, such as 10.1.79.1

Filename — the file that you are transferring, such as abcfile.txt

USER—A user name used for authentication

PASSWORD — a password used for authentication

MODE—always binary

Password variable—The password variable you want to use to obtain password authentication.

Examples

USER and PASSWD:

```
1 ftp://10.1.79.1/abcfile.txt <USER=ann PASSWD=ann@somewebsite.com>  
<MODE=binary>
```

USER_VAR and PASSWD_VAR:

```
1 ftp://10.1.79.1/abcfile.txt <USER_VAR=<APPLY groupnames>  
PASSWD_VAR=<APPLY passwords>> <MODE=binary
```

NOTE: To use the APPLY command you must first ASSIGN a run-time value to the variable

To create an Actions list for an FTP test with Reflector or SmartBits (Device test):

If you are using Reflector or a SmartBits module as the FTP server, you can access a virtual file system that consists of any size file. The size is determined by the URL's postfix in the FTP Action list. For example

1b: a file with a size of 1 byte

10b: a file with a size of 10 bytes

100b: a file with a size of a 100 bytes

1k: a file with a size of 1 Kbytes

10k: a file with a size of 10 Kbytes

100k: a file with a size of 100 Kbytes

1m: a file with a size of 1 Mbyte

10m: a file with a size of 10 Mbytes

100m: a file with a size of a 100 Mbytes

1g: a file with a size of 1 Gigabyte.

Syntax

```
1 ftp://server_IP_address/file_size
```

Server IP address—The address of the server, such as 10.1.79.1

File size—size of the file, such as 1b for 1 byte

Example

For a file size of 1 Gigabyte

```
1 ftp://192.168.42.11/1g
```


The above example uses an IPv4 address. The following example uses an IPv6 address:

```
1 ftp://[2106::0200:FF:FE00:8101]/1g
```

Defining the Server Profile

Complete this section only if you are defining a Device test.

To create an FTP Test:


1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the server profile, and then select **FTP** as the server type.
4. Enter the FTP server port. The default is 21.

IMPORTANT: If the TCP Inactivity Timer in the Server Network tab is less than two seconds, the FTP control connection could be aborted. Otherwise, the FTP application has a work-around to send a Keep Alive packet over the control connection. The data connection will not be idle, and therefore, is unaffected by the TCP Inactivity Timer.

Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

Testing HTTP

To complete information and run the test:

1. Learn about Avalanche HTTP testing.
2. Add an Actions list.
3. Define information for the client.
4. Define information for the server.(Device test only)
5. Define server transaction objects. Device test only)
6. Run the test and review results.

TIP: See HTTP example test scenarios in the Help for the client and server behavior that is a result of various client/server parameter settings.

About HTTP Testing

Avalanche uses HTTP to request a series of objects from a server, or to submit forms databases by using embedded strings. To accomplish this, an HTTP Action list uses three elements: Level, Method, and URI. Sometimes the URI is appended with an embedded string that affects client-server exchanges. In addition to using an Action list, Avalanche provides a series of parameters that you can use to configure the client profile. For example, you can set user behavior, the transport mode, browser emulation, and SSL configuration.


Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom. You can create an Action list for Application tests or for Device tests. In addition to defining HTTP Level, Method, and URI, Avalanche supports additional HTTP actions that you can use to perform a variety of other tasks such as searching and assigning variables. Actions include:

- HTTP Match and Match Not
- HTTP Stop
- HTTP Search
- HTTP Assign Variables
- HTTP Apply
- HTTP Dynamic URL
- HTTP Set Runtime Cookie Action
- HTTP Additional Headers with Variables
- HTTP URL Encode
- HTTP Response Body Decompress

For more information about these actions including syntax and examples, see the associated topic in the Help.

To create an Actions list:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.
3. Enter a description for the Actions list.
4. Enter information in the Action list as defined by the following syntax.

Syntax

The URI states the protocol, the server, and the requested object. HTTP Action list entries must state the protocol (http://) on each line.

```
1 or 2 method HTTP:// server ip address/requested object
```

Examples

GET method:

```
1 GET http://10.10.10.10/
1 GET http://192.168.44.1/Transaction_1
1 GET http://www.somewebsite.com/
2 GET http://www.somewebsite.com/images/about.gif
```

HEAD method:

```
1 HEAD http://www.somewebsite.com/support/support.shtml
2 HEAD http://www.somewebsite.com/images/logo1.gif
```

POST method:


```
1 POST http://www.somewebsite.com/ username=<filename.$1>
password=<filename.$2>
NEXT ROW <filename>
1 POST http://www.somewebsite.com/cgi-bin/ username=<filename.$1>
password=<purchase.$1>
```

Defining Client Information

Define the client HTTP parameters that you want to use. See the Help for more information about the fields in a Client Profile pane.

TIP: If you want to use client default values, you can skip the following procedure.

To configure the client:

1. Click the **Client** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile or select the profile with which you want to work.

3. Enter a description for the server profile, and then select **HTTP** as the server type.
4. In the User Behavior pane, enter information to define a profile that simulates how a user behaves, such as the amount of time a user spends on a page at a site, or to simulate users logging in to a site. You also use this pane to create and associate different files with a test. For example, you can:
 - o create search criteria files for use with an HTTP Search, Match, or Match Not action.
 - o create HTTP Body content files that you can send by using an HTTP POST command.
 - o create forms databases that you can use to send information during a test. You can use a forms database to simulate users submitting GET and POST requests to an Internet site.
 - o create DNS host mapping files for mapping host names to IP addresses in an Action list.
5. Complete entries in the Browser Emulation and Protocol panes to set parameters that determine how simulated users communicate with the targeted network.


If you select a standard browser such as Microsoft Internet Explorer or Netscape Navigator, default protocol entries appear in the Protocol Level and Browser Header panes.

If you select User Defined browser, the fields in the Browser Header pane become available so that you can define header strings.
6. To test Basic or Proxy Authentication, complete entries in the Authorization pane.

Defining the Server Profile

Complete this section only if you are defining a Device test and if you want to change default values.

To define a server profile:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the test, and then select **HTTP** as the server type.
4. Enter the port where the server resides. The default port is 80.
5. Make selections in the Connection Properties pane, such as connection termination. You can select an item from the Transaction Profile drop-down menu as the object that the server responds with by default, if no object is included in the request on the Client Actions list. (See the next section "Defining Server Transaction Objects.")

6. Complete entries in the Server Emulation pane, such as server type, transaction profile, and protocol level.
7. To improve performance, select options in the HTTP Streamliner Mode pane, if compatible with your test.

NOTE: The fields in the HTTP Streamliner Mode pane are activated only if you select the Enable HTTP Streamlined Mode checkbox in the Run Configure tab. This feature provides a higher performance HTTP transaction mode on a test-wide basis. A pre-built HTTP response, derived from the default transaction profile for the server, is used to achieve the higher performance.

8. To enable cookie support, select **Cookies**, and enter the settings that you want to use.

Defining Server Transaction Objects

Complete this section only if you are defining a Device test. You use the **Server Transactions** tab to configure the HTTP/HTTPS server transactions for tests. To specify an alternate (or custom) Transaction Profile in a URL list, see Customizing HTTP/HTTPS Responses in the Help.


To define a transaction object:

1. Click the **Server** tab, and then the **Transactions** tab.
Select the default transaction from the Transaction Profile drop-down menu.
2. Select the options on the Server Transactions tab that you want to use. The data body types and body sizes are especially important for your test.

Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

Testing MM4

To complete MM4-specific information and run the test:

1. Learn about MM4 testing
2. Add an Actions list
3. Define the client profile
4. Define the server profile
5. Run the test and review results

About MM4 Testing

With Multimedia Messaging Service (MMS), cell phone providers can send and receive multimedia messages. MM4 provides communication between the Multimedia Messaging Service Relays/Servers, also referred to as MMS Centers (MMSCs). The sender is the originator MMSC, and the receiver is the recipient MMSC.

By configuring an MM4 test, you can use Avalanche to emulate an originator MMS Relay/Server, simulating the forwarding of MM4 messages to another MMS Relay/Server, and use Reflector to emulate a recipient MMS Relay/Server. For more information about the role of Avalanche and Reflector, see Emulating MM4 in the Help.

NOTE: This topic uses Avalanche and Reflector as examples of a client and server to explain the MM4 test, however, you can alternatively use your own client and server devices.

Adding an Actions List

An Action list defines the addresses, requests, and message for the test. The format of the MM4 Actions list is similar to an SMTP Action list. You identify an envelope (FROM and TO information), a subject, and a message.

However, MM4 Action lists allow for three additional optional parameters—ACK_FORWARD, DELIVERY_REPORT, and READ_REPLY—to specify response and request messages.

Syntax

```
mm4://mmsc_ip_address_relay_server FROM=<sender_address>  
TO=<recipient_address> [SUBJECT=<subject>] DATA=<FIXED size> [ACK_FORWARD][  
DELIVERY_REPORT] [READ_REPLY]
```

NOTE: Use a DATA or MM4_BODY_FILE tag to define the message that you want to send. You cannot use DATA and BODY tags at the same time, however you must use one to indicate the data for the message. You can use MM4_ATTACH_FILE optionally with MM4_BODY_FILE.

mmsc_ip_address_relay_server—Identifies the IP address of the intended MMS Relay/Server. (required)

FROM=sender_address—Specifies the sender's (originator's) email address and domain name. (required)

TO=recipient_address—Specifies the recipient (terminator's) email address and domain name. You can identify multiple addresses, separating each address with a comma or a space, such as Joe@somewebsite.com, Mary@somewebsite.com, or Joe@somewebsite.com Mary@somewebsite.com. (required)

SUBJECT=subject—Text that specifies the subject information. The subject content cannot contain spaces unless you include them within quotation marks, such as "". If you don't enter a subject, a randomly generated subject is used for the test. (optional)

DATA—Specifies the content of the message sent. (One of the following is required.)

DATA=<FIXED size> An auto-generated message body that is the size that you indicate here. This length does not include the message header.

DATA=<RANDOM UNIFORM min_size max_size> An auto-generated message body that is randomly generated to be between the minimum and maximum sizes that you indicate here.

MM4_BODY_FILE=<"filename" "content type"> The filename and content type of the body that is included as the MIME (Multipurpose Internet Mail Extensions) part that represents the body of the message, such as the body of an email message. Format the contents of the file that you specify for the MM4_BODY_FILE as ASCII text, since it is used as is.

MM4_ATTACH_FILE=<"filename" "content type"> (You can use this tag optionally with MM4_BODY_FILE.) The filename and content type of an attachment that is included in the MIME message, such as an attachment that you would add to an email message. The contents of the file that you specify for an MM4_ATTACH_FILE will be base64-encoded. You can use the parameter multiple times to attach multiple files. You can specify any content type, such as plain text or image. If you use this parameter, you must also use the MM4_BODY_FILE parameter.

IMPORTANT: You must add the files that you specify as BODY or ATTACH files as content files in the Content Files Tab before you run the test.

ACK_FORWARD—Requests that the recipient (such as Reflector) send a forward response (MM4_forward.RES). (optional)

DELIVERY_REPORT—Requests that the recipient (such as Reflector) send a delivery_reportrequest (MM4_delivery_report.REQ) (optional)

READ_REPLY — Requests that the recipient (such as Reflector) send a read_reply_reportrequest (MM4_read_reply_report.REQ). (optional)

NOTES: If you include the three previous parameters in an Action list, you can additionally control the percentage of requests and responses from the server by setting values in the MM4 percentage fields on the Server Profiles tab. Using these fields you can choose to deny whatever

percentage of the requests/responses you want. For example, to deny all requests, you can set the value to 100% denial. For a diagram showing example requests and responses sent between Avalanche and Reflector, see Emulating MM4.

Use the MM4 fields in the Client and Server Profile tabs to set minimum and maximum time-out values for the MM4 transactions and session.

Examples

The following example sends an MM4_forward_REQ message with a 100-byte body from one user to another user. The recipient is requested to respond with an MM4_forward.RES, MM4_delivery_report.REQ, and MM4_read_reply_report.REQ messages.

```
mm4://1.2.3.4 FROM=<user1@somewebsite1.com>  
TO=<user2@somewebsite2.com> SUBJECT=<my_subject> DATA=<FIXED  
100> ACK_FORWARD DELIVERY_REPORT READ_REPLY
```

The following example sends an MM4_forward_REQ message with a body file and three attached files from one user to another user. The recipient is requested to respond with an MM4_forward.RES. Other requests and responses are not required.

```
mm4://1.2.3.4 FROM=<user1@somewebsite1.com>  
TO=<user2@somewebsite2.com> SUBJECT=<my_subject>  
MM4_BODY_FILE=<"mymsg" "plain/text">  
  
MM4_ATTACH_FILE=<"my_attacheddoc" "plain/text">  
  
MM4_ATTACH_FILE=<"my_picture" "image/jpg">  
  
MM4_ATTACH_FILE=<"my_audio" "audio/amr">  
  
ACK_FORWARD
```

Defining the Client Profile

Use the MM4 Configuration fields on the Client Profiles tab to define information to emulate an originator MMS Relay/Server, simulating the forwarding of MM4 messages to another MMS Relay/Server. Each MM4 message is sent by using a separate SMTP session on a client such as Avalanche, and a server such as Reflector.

To define a client profile:


1. Click the **Client** tab, and then the **Profiles** tab.
2. Select the client profile with which you want to work.
3. In the MM4 Configuration pane, enter the originator port, response timeout, and session timeout. See the Client Profiles MM4 Help for detailed definitions.

Defining the Server Profile

Use the MM4 fields on the Server Profile tab to emulate a recipient MMS Relay/Server. Each MM4 message is sent by using a separate SMTP session on

a client such as Avalanche, and a server such as Reflector. For detailed definitions of the server fields, see Server Profiles Tab MM4 in the Help.


To define a server profile:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the test, and then select **MM4** as the server type.
4. Enter the port where the server resides. The default port is 25.
5. Enter information to define the MM4 settings, including the following:
 - o MM4 emulation to define timeouts and percentage of report request denied and acknowledged.
 - o Request status to define the percentage of each report status response that is randomly sent by the server as an MM4_forward.RES.
 - o Message status to define the percentage of each request status that is randomly sent as an MM4_delivery_report.REQ.
 - o Read status to define the percentage of each read status that is randomly sent by the server as an MM4_read_reply_report.REQ.

Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

Testing POP3

To complete POP3-specific information and run the test:

1. Learn about Avalanche POP3 testing.
2. Add an Actions list.
3. Define the server profile. (Device test only)
4. Run the test and review results.

About POP3 Testing


A POP3 test uses Avalanche to simulate a client logging into a server, executing supported commands, fetching various mail messages from a POP3 server, and then closing the session. Reflector or a SmartBits module can emulate a POP3 Server and respond to POP3 commands generated by Avalanche for POP3 traffic, allowing you to stress test network devices that are aware of mail traffic.

The Action list describes the transactions and controls their sequence of execution. By default, POP3 uses port 110 for transport, and levels are not relevant to the POP3 protocol.

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom.

To add an Actions list for a POP3 test:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.

Enter information that simulates a POP3 client sending information. Refer to the following syntax and examples.

Syntax

```
pop3://IP_Address USER PASSWD command
```

If you want to use a forms database to supply username and password

```
pop3://IP_Address USER=filename.$1 PASSWD=filename.$2 command
```

IP_Address-Identifies the web address of the test server.

USER-Specifies the users name. Must match mailbox name configured in Server Profiles tab. If using a forms database, specifies the filename.

PASSWD-Specifies the user's password. Must match mailbox name configured in Server Profiles tab. If using a forms database, specifies the filename.

command-Avalanche supports the following commands:

RETR- Retrieves all the messages from the server. Precede this command by the CHECK command.

CHECK, CHECK = integer-Checks the number of messages on the server. If an integer follows the command, Avalanche checks that the integer is equal to the number of messages on the server. This is equivalent to the POP3 STAT command.

LIST-Issues the equivalent of the POP3 LIST command. The server should issue a multi-line response for each message on the server.

LIST=UIDL-Issues the equivalent of the POP3 UIDL command. The server should issue a multi-line response for each message's unique identifier.

DEL (or DELETE)-Deletes all the messages on the server. Precede this command by the CHECK command. If used in conjunction with the RETR command, the RETR command should precede the DEL command.

Examples

Action list line entries for POP3

```
POP3://192.168.44.11 USER=admin PASSWD=admin CHECK LIST
```

```
POP3://www.somewebsite.com USER=admin PASSWD=admin CHECK LIST
```

Action list for forms database access for POP3

```
POP3://192.168.44.11 USER=pop3db.$1 PASSWD=pop3db.$2 CHECK  
RETRIEVE
```

NOTE: USER and PASSWD names must be the same and must match the mailbox name configured in the Server Profiles tab.

An example forms database supporting the previous Action is as follows:

```
username1,username1  
username2,username2  
username3,username3  
username4,username4  
username5,username5  
username6,username6  
username7,username7  
username8,username8  
username9,username9  
username10,username10
```

An IPv4 address

```
POP3://192.168.44.11 USER=admin PASSWD=admin CHECK LIST
```



An IPv6 address

```
POP3://[2107::0200:FF:FE00:8202] USER=admin PASSWD=admin  
CHECK LIST
```

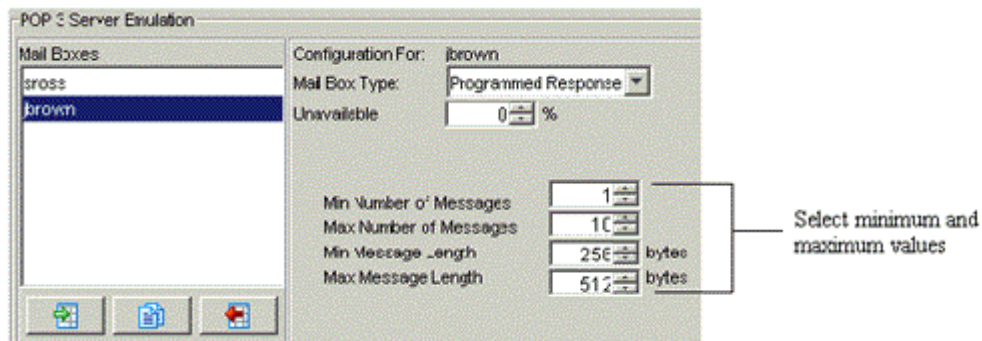
Defining the Server Profile

Complete this section only if you are defining a Device test. Use a server profile to specify the type of messages sent to a mailbox and the mailbox name. The USER and PASSWD names identified in the Actions list must match the mailbox name you identify in the Server Profiles tab.

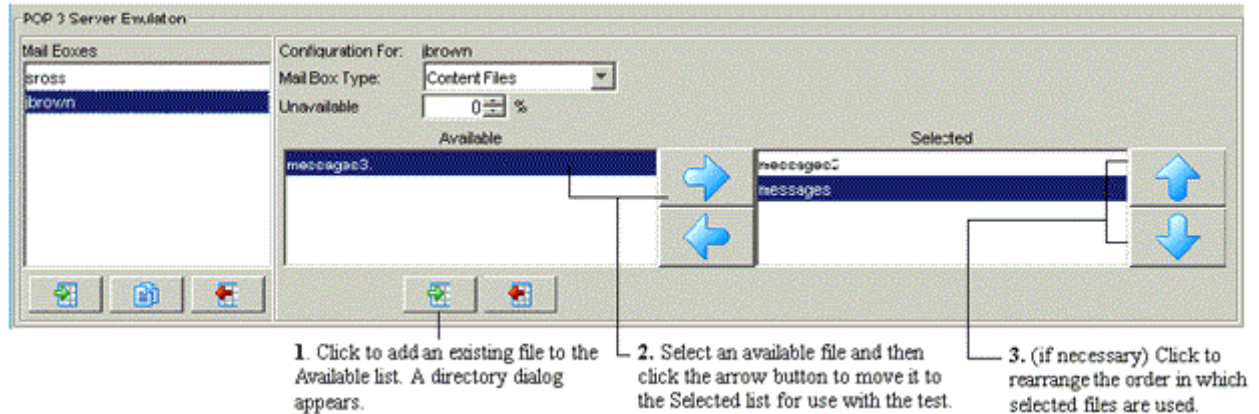
To define an SMTP server profile:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the profile, and then select **POP3** as the server type.
4. Enter the port where the server resides. The default port is 110.
5. Click the **Add Mailbox** button  to add a new mailbox.
6. Enter a unique, alphanumeric mailbox name. The only special characters you can use are dash (-), underscore (_), and period (.). The mailbox name and password associated with the mailbox are the same.
7. Select either **Programmed Response** or **Content Files** as the mailbox type, and then complete configuration information:

Programmed Response specifies a range of messages and message lengths. Each time there is a connection to a mailbox, Avalanche sends a random number of simulated POP3 messages within the specified range and size.




Content Files specifies POP3 message files. Each time you connect to the mailbox, Avalanche sends a POP3 message file that you have created and identified. Do not include the following characters in a POP3 message file: <, >, \, or ;.



Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

NOTE: The statistical data for POP3 tests are described in POP3 Statistics in the Help.

Testing RTSP/RTP Streaming

To complete RTSP/RTP-specific information and run the test:

1. Learn about Avalanche RTSP/RTP testing.
2. Add content files. (Device test only)
3. Add an Actions list.
4. Define the server profile. (Device test only)
5. Run the test and review results.

About RTSP/RTP Testing

Avalanche emulates various clients that use RTSP/RTP to retrieve streaming media files. Essentially, a simulated user requests a specific server to transmit a specific file. If you use Reflector or a SmartBits module, it emulates a streaming server, establishing a connection with a client, returning QuickTime streaming media, Windows Media, or MPEG files, and then terminating the session.

Adding Content Files

If you run a test against an actual streaming server, you do not need to add a content file. To run a test with Avalanche and Reflector, or SmartBits modules, load the specified file as a content file before you start the test. The files include the following:

cawsample_36k_60s_va.mov.caw: A 60-second stream file encoded at a 36K rate that has both video and audio channels.


cawsample_80k_378s_vo.mov.caw: A 378-second stream file encoded at an 80k rate that has only a video channel.

cawsample_160k_60s_va.mov.caw: A 60-second stream file encoded at a 160k rate that has both video and audio channels.

cawsample_250k_60s_va.mov.caw: A 60-second stream file encoded at a 250k rate that has both video and audio channels.

NOTE: The four different .mov streaming files are annotated, have different encoding rates and time spans, and can only be streamed by Reflector or SmartBits. Reflector and SmartBits will NOT work with regular .mov files. Complete the steps in this section only if you are defining a Device test.

To add content files:


1. Click the **Content Files** tab.
2. Click the **Add Content File** button . A directory dialog box appears. By default, it lists files that appear in the default Content file directory. When you first install the Avalanche software, this directory contains the sample streaming files described previously.
3. Select the file, and then click the **Add** button. The file appears in the **Content File** tab.

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user sending a request to your test server. As Avalanche simulates new users, each one uses the list from top to bottom. An RTSP Action list uses two elements: Level and URI. Enter a 1 preceding the URI to signify a top-level retrieval.

NOTE: The Ramp Down phase time in the load profile must have enough time to allow all the streams to finish playing; otherwise, uncompleted sessions are counted as failures. See the Loads Tab in the help for more information. It is recommended that the Ramp Down phase is at least as long as the time required for the longest streaming file that you are requesting.

To add an Actions list for an RTSP/RTP test:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.

Enter information that simulates retrieving streaming media. Refer to the following syntax and examples.

Syntax

```
1 RTSP://server_IP_address/directory/filename.mov  
1 RTSP://server_IP_address/filename.mov
```

Server IP address—The address of the server, such as 10.1.79.1.

Directory—The directory where the file is stored. Define this information if you are testing against an actual server, using an Application test.

NOTE: Most commercially available streaming servers have a default directory for content files. If the file you are requesting is in that directory, you do not need to add the full path to the Action list.

Filename—The name of the streaming file.

Examples

Using an actual server

```
1 RTSP://10.1.79.1/movies/thriller.mov  
1 RTSP://10.1.79.2/movies/drama.mov
```

Using Reflector or server SmartBits modules


```
1 RTSP://10.1.79.2/cawsample_36k_60s_va.mov
```

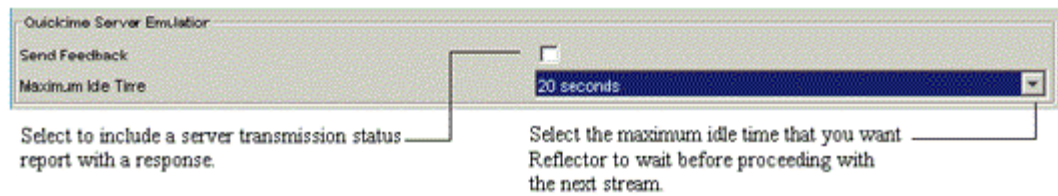
IMPORTANT: For the Reflector/SmartBits files, the actual file name (as shown in the Content tab) includes the extension .caw, such as cawsample_36k_60s_va.mov.caw. However, when requesting a file in an Actions list, do not include the .caw. For example, identify the file as cawsample_36k_60s_va.mov.

Defining the Server Profile

Complete this section only if you are defining a Device test. Use a server profile to define the port and server emulation.

To define a server profile:


1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the test, and then select **Streaming** as the server type.
4. Enter the port where the server resides. The default port is 554.
5. Select emulation settings.



Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

Testing SIP

To complete SIP information and run the test:

1. Learn about SIP testing.
2. Define information for the client.
3. Add an Actions list.
4. Add a content file (if you want to use pre-recorded Wave files).
5. Define information for the server. (Device test only)
6. Run the test and review results.

About SIP Testing

SIP is an application-layer control protocol that can establish, modify, and terminate multimedia sessions (conferences) such as Internet telephony calls. RTP data streams carry the voice data.

Avalanche performs the role of multiple clients. It initiates a SIP session by using the corresponding SIP message exchange, sends pre-recorded Wave files, receives RTP data, and terminates the session by using the corresponding SIP message exchange. Avalanche supports SIP over UDP and TCP transport modes. Avalanche processes any session update messages it receives, but does not initiate session update messages. Avalanche can generate calls to real SIP devices, such as IP Phones, and with Reflector, can transmit traffic between gateways, SIP firewalls and SIP proxy servers. For more information about the role of Avalanche, see Configuring SIP in the Help.

You can use Reflector to perform the role of multiple SIP user agents, terminating the call. Reflector accepts incoming calls, and echoes the RTP stream, if any. Reflector processes any session update messages it receives, but does not initiate session update messages. Reflector can accept calls from real SIP devices, such as gateways, SIP firewalls, SIP phones, and SIP proxy servers. For more information about the role of Reflector and detailed Server Profile field definitions, see Server Profiles Tab SIP in the Help.

Defining Client Information

Define the client SIP parameters that you want to use.

To configure the client:


1. Click the **Client** tab, and then the **Profiles** tab.
2. Select the client profile with which you want to work.
3. In the SIP Configuration pane, complete entries to define the local SIP port, the port number on which the first RTP listeners will be created, and the number of RTP channels that will be created. Enter a port number value that is even and in the acceptable port range. The port value is incremented by two for each additional RTP channel. It is recommended that you set the number of channels to be greater than the expected maximum number of simultaneous calls to provide more

realistic traffic. For more information, see Client Profiles SIP Fields in the Help.

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user sending a request to your test server. As Avalanche simulates new users, each one uses the list from top to bottom.

To add an Actions list for a SIP test:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.

Enter information that simulates SIP. Refer to the following syntax and examples.

Default Syntax

```
sip://192.168.44.1 LOCALHOST="caw.com" TRANSPORT=UDP
```

LOCALHOST—The host name used in the From header.

TRANSPORT—The transport mode used in the session, in this case UDP.

NOTE: The default syntax does not specify a Wave file. Without a Wave file, Avalanche sends an on-hold Session Description Protocol (SDP); Avalanche does not send RTP data during the session. If the URL has a Wave file parameter, SDP is constructed based on the Wave file codec. After session setup, Avalanche starts sending the stream from the Wave file.

In addition, because the default syntax does not specify call length parameters in the URL, Avalanche defaults to a call length randomly chosen between a minimum of 10000 ms (10 seconds) and a maximum of 30000 ms (30 seconds). As soon as the call is accepted, Avalanche calculates the call time, and terminates the call after this time.

Syntax

```
sip://ip:[port]
LOCALHOST = STRING
[REMOTEUSER = STRING]
[REMOTEHOST = STRING]
[CALLLENGTH_MIN = INTEGER]
[CALLLENGTH_MAX = INTEGER]
[TRANSPORT = (UDP | TCP)]
[LOCALUSER = STRING]
[WAVE = STRING]
[MAXFORWARDS = INTEGER]
```

[REQUESTURI = STRING]

[SIPHEADER = STRING]

LOCALHOST-The host name used in the From header.

REMOTEUSER-The username used in the To header. If no request URI is configured, this name will be used in the Request URI.

REMOTEHOST-The host name used as host in the To header. If no REQUESTURI is configured, this name will be used in the Request URI. By default, the destination host of the Action is used.

CALLLENGTH_MIN, CALLLENGTH_MAX-The number of milliseconds used as the interval (between MIN and MAX) for the random selection of the call time. The default MIN is 10000 (10 seconds) and the default MAX is 30000 (30 seconds).

TRANSPORT-The transport mode used in the session: UDP (default) or TCP.

LOCALUSER-The username used in the From and local Contact header.

WAVE-The name of the Wave file. If present, the client will send the RTP data for the session from this file.

MAXFORWARDS-The value of the Max-Forwards header. The default is 70.

REQUESTURI-If present, used as the Request URI in the initial request. Otherwise, the default sequence as described for REMOTEHOST is used.

SIPHEADER-"Header name: value" adds an additional header to the first INVITE request. One Action can have several SIPHEADER parameters.

NOTES: The host and port (default 5060) in the URI string are used as the destination for the first request in the call for UDP, and all requests in the call for TCP. The destination of requests inside the call for UDP depends on the received Contact header and Route set.

To add a predefined route set for the first request in the call for UDP, and all requests for TCP, the REQUESTURI must be configured to have the correct value according to RFC 3261, and the set of Route headers must be added by using the SIPHEADER parameter of the Action.

Examples

A minimal SIP URL:

```
sip://192.168.42.11 LOCALHOST="somewebsite.com"
```

The LOCALHOST parameter is used as the host part of the From header in SIP messages. This is the only mandatory parameter.

If the remote port differs from 5060, it should be specified. In the following example, 5067 is used:

```
sip://192.168.42.11:5067 LOCALHOST="somewebsite.com"
```

NOTE: If the IP address or port is incorrect, the session is considered unsuccessful after 32 seconds for UDP. This is a protocol-specific feature.

WAVE parameter:

```
sip://192.168.42.11 LOCALHOST="somewebsite.com" WAVE
="welcome.wav"
```

The WAVE parameter specifies a Wave file name to be used. If this parameter is not specified, the session will be a signaling-only session without RTP.

NOTE: You must upload the corresponding Wave file using the Content Files tab.

The current Avalanche release supports two types of Wave file data encodings. The codec for RTP data is based on the Wave file data encoding. The following is a list of supported Wave data encodings and corresponding RTP data codecs:

```
CCITT u-Law 8, 000 kHz, 8-bit, mono-G711.U codec is used
CCITT A-Law 8, 000 kHz, 8-bit, mono-G711.A codec is used
```

NOTE: These encodings are supported by most sound recorders, including a standard Windows sound recorder.

If the file length is greater than the call length, the file is truncated. If the file length is less than the call length, the file is replayed.

TRANSPORT parameter:

The TRANSPORT parameter can be either UDP or TCP. If this parameter is omitted, UDP will be used as the default transport mode.

NOTE: This is the only method to specify the transport mode on the client side.

UDP examples:

```
sip://192.168.42.11:5067 LOCALHOST="somewebsite.com"
sip://192.168.42.11:5067 LOCALHOST="somewebsite.com"
TRANSPORT=UDP
```

TCP example:

```
sip://192.168.42.11:5067 LOCALHOST="somewebsite.com"
TRANSPORT=TCP
```

CALLLENGTH_MIN and CALLLENGTH_MAX parameters:

```
sip://192.168.42.11:5067 LOCALHOST="somewebsite.com"
CALLLENGTH_MIN = 20000 CALLLENGTH_MAX = 30000
```

The call length is randomly selected from the interval between CALLLENGTH_MIN and CALLLENGTH_MAX. The values are in milliseconds. The call length in this example will be between 20 and 30 seconds. The CALLLENGTH_MAX must be less than the call timeout value on the server side.

REMOTEUSER parameter with a phone number:


```
sip://192.168.42.11:5067 LOCALHOST="somewebsite.com"  
REMOTEUSER="18007747368"
```

The REMOTEUSER parameter specifies the username or dial number of the remote site. It is not mandatory when Reflector accepts the call, but may be required by a real device.

Adding a Content File

If you configured the Actions list to use pre-recorded Wave files, upload them by using the Content Files tab before you run the SIP test.


To add content files:

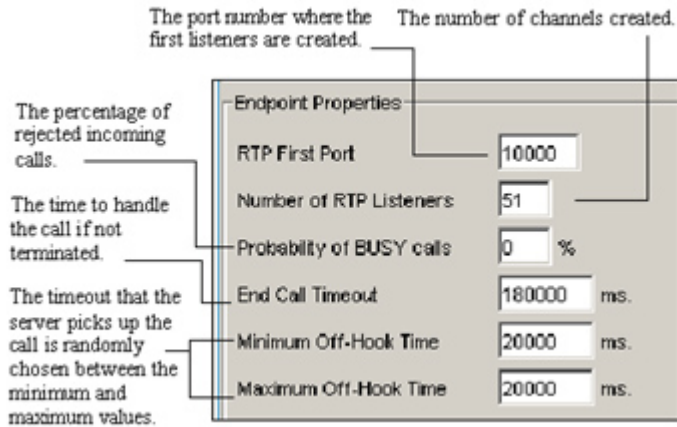
1. Click the **Content Files** tab.
2. Click the **Add Content File** button . A directory dialog box appears. By default, it lists files that appear in the default Content file directory. When you first install the Avalanche software, this directory contains sample streaming files. You can also store your own content files in this directory.
3. If you stored your files in the default directory, skip to the next step. Otherwise, navigate to the location that contains the Wave file that you want to use.
4. Select the file, and then click the **Add** button. The file appears in the Content Files tab.

Defining the Server Profile

Complete this section only if you are defining a Device test. Use the **Server Profiles** tab to configure information for a SIP server.

To define a server profile:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the profile, and then select **SIPTCP** or **SIPUDP** as the server type.
4. Enter information in the Endpoint Properties fields.




NOTE: Before the Off-Hook time expires, the ringing signal is sent to the caller.

Running the Test and Reviewing Results

After completing all set-up steps, run the test.

NOTE: For information about SIP statistical data see the client SIP Statistics and server SIP Statistics in the Help.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

Testing SMTP

To complete SMTP-specific information and run the test:

1. Learn about Avalanche SMTP testing.
2. Add an Actions list.
3. Add content files (If you are attaching files)
4. Define the server profile. (Device test only)
5. Run the test and review results.

About SMTP Testing

Avalanche simulates a client sending various mail messages from an SMTP server. The Action list describes the list of transactions and controls their sequence of execution. By default, SMTP uses port 25 for transport, and levels are not relevant to the SMTP protocol.

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom.


NOTE: You can alternatively use ESMTP actions instead of SMTP actions in your Action list. For more information, see ESMTP Actions.

With an SMTP Action list you define:

Envelope information: From, To, and Subject (optional).

Message information: The content of the message. You can use several ways to provide the message.

To create an Actions list for an SMTP test:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.
3. Enter actions that simulate SMTP. Refer to the following syntax and examples.

Syntax

The basic syntax is:

```
smtp://IP_Address FROM=<user@host> TO=<user@host>  
SUBJECT=<"my_subject"> DATA=<content of message>
```

See the following syntax for details about each component, followed by an explanation of each component and examples of use.

Specifying the sender (FROM)

```
FROM=<user@host>
```

or

```
FROM=<database^#>
```

NOTE: The above syntax specifies a forms database, where *database* is the filename and # is the column number.

Specifying the recipient (TO)

```
TO=<user@host>
```

or

```
TO=<database^#>
```

You can specify the recipient as a list of email addresses in comma-separated value (CSV) format. For example,

```
TO=<user@host,user@host>
```

or a combination of email addresses and addresses in a forms database

```
TO=<user@host,database^#>
```

Specifying the subject (SUBJECT) (optional)

```
SUBJECT=<databasefile^#>
```

or

```
SUBJECT=<"this_is_my_subject">
```

Sending data

```
DATA=<content of message>
```

Sending a body file

```
SMTP_BODY_FILE=<"filename" "content-type">
```

Sending a raw message

```
DATA=<FIXED, bytes> SMTP_RAW_MESSAGE=<"filename">
```

Using a forms database with fixed content

```
DATA=<FIXED,300>
```

IP address-Identifies the IP address of the test server.

FROM-Specifies the sender information, using its email address and domain name.

TO-Specifies the recipient.

SUBJECT (optional)-Specifies the subject information. You can include a subject or reference a forms database file. If you don't specify SUBJECT, Commander uses a random subject during the test. The subject content cannot contain spaces. Keywords are not case-sensitive.

DATA (optional if using a body, attach, or raw file)-Specifies the content of the email message sent to the SMTP server. With DATA, you can use the following parameters:

FIXED-Followed by an integer, such as FIXED 300, specifies the length of the generated string used as the message body. This length does not include the message header.

RANDOM=UNIFORM- Followed by two integers, such as RANDOM=UNIFORM 300 600, specifies a randomly generated string of random length to be used as the message body. The length is uniformly distributed between a minimum and a maximum set of values.

SMTP_BODY_FILE (optional)-As an alternative to DATA, you can specify the filename and content-type of the body that should be included as the Multipurpose Internet Mail Extensions (MIME) that represents the body of the message. Format the contents of the file that you specify for the SMTP_BODY_FILE as ASCII text, since it is used as is.

SMTP_ATTACH_FILE (optional)-Specifies the filename and content-type of an attachment to include as an attachment in the MIME Message. The contents of the file that you specify for an SMTP_ATTACH_FILE is base64-encoded. You can specify any text for content-type; it is appended to the Content-Type header.

SMTP_RAW_MESSAGE (optional)-Specifies the raw data's filename of the entire MIME message included after the standard headers. The contents of the file that you specify for an SMTP_RAW_MESSAGE can be in any format, and is used as is. If you use SMTP_RAW_MESSAGE, you cannot use SMTP_BODY_FILE or SMTP_ATTACH_FILE.

IMPORTANT: You must add the files that you specify as content files in the Content Files Tab. See Adding Content Files later in this topic.

Actions (optional)-You can specify any number of the following to be sent: RSET, EHLO (using ESMTP), SOML, SEND, SAML, VRFY, EXPN, HELP, TURN, ETRN, and XXXX.

NOTE: If you use Reflector or a SmartBits module as the server, there is no decode logic to act on these Actions, simply an acknowledgement that the Action was sent.

REPEAT (optional)-Specifies the number of times to send the message within the same SMTP session. The default value is 1.

DATABASE-Identifies the filename and column number (#) if you use a forms database.

Examples

IP Address Examples:

An IPv4 address

```
smtp://192.168.42.11 FROM=<a@somewebsitea.com>  
TO=<b@somewebsiteb.com> DATA=<FIXED,300>
```

An IPv6 address

```
smtp://[2107::0200:FF:FE00:8201] FROM=<a@somewebsitea.com>  
TO=<b@somewebsiteb.com> DATA=<FIXED,300>
```

Simulated content:

Fixed 300 bytes

```
smtp://192.168.42.14 FROM=<user1@somewebsite.com>  
TO=<user2@somewebsite.com> SUBJECT=<TEST> DATA=<FIXED, 300>
```

Random 300-600 bytes

```
smtp://192.168.42.14 FROM=<user1@somewebsite.com>  
TO=<user2@somewebsite.com> SUBJECT=<TEST>  
DATA=<RANDOM=UNIFORM 300 600>
```

Content in body file:

```
smtp://192.168.42.14 FROM=<user1@somewebsite.com>  
TO=<user2@somewebsite.com> SUBJECT=<TEST>  
SMTP_BODY_FILE=<"mymsg" "plain/text">
```

Multiple recipients, fixed data content, SOML to be sent, and the message repeated three times:

```
smtp://192.168.42.11 FROM=<a@somewebsitea.com>  
to=<b@somewebsiteb.com,c@somewebsitec.com> DATA=<FIXED,300> SOML  
REPEAT=3
```

Fixed content with raw virus file:

```
smtp://192.168.42.14 FROM=<user1@somewebsite.com>  
TO=<user2@somewebsite.com> SUBJECT=<I love you>  
DATA=<FIXED, 300>  
SMTP_RAW_MESSAGE=<"iloveyou.exe">
```

Raw data file as the entire MIME message when the file does not need to be encoded:

```
smtp://192.168.42.11 FROM=<a@somewebsitea.com>  
to=<b@somewebsiteb.com> subject=<"my_subject">  
SMTP_RAW_MESSAGE=<"themsg">
```

File as the message body and other files attached:

```
smtp://192.168.42.11 FROM=<a@somewebsitea.com> to=<b@somewebsiteb.com>  
SMTP_BODY_FILE=<"mymsg" "plain/text">  
SMTP_ATTACH_FILE=<"wordDoc" "application/msword">  
SMTP_ATTACH_FILE=<"PDFDoc" "application/pdf">  
SMTP_ATTACH_FILE=<"ExcelDoc" "application/vnd.ms-excel">
```

NOTE: If you use SMTP attachments, consider that large attachment files create extensive network overhead. Some error conditions may occur, such as throttling due to low memory, if the total size of Fixed content with forms database:

```
smtp://192.168.42.11 from=<SMTPdb^1> to=<SMTPdb^2> subject=<SMTPdb^3>  
DATA=<FIXED,300>
```


The following forms database could support the previous forms database action:

```
asmith@somewebsite.com, eyee@somewebsite.com, Hi
bbrown@somewebsite.com, jsant@somewebsite.com, Hi
cjones@somewebsite.com, rmoe@somewebsite.com, Hi
dkline@somewebsite.com, ejay@somewebsite.com, Hi
```

Adding Content Files

If you want to use SMTP_BODY_FILE, SMTP_ATTACH_FILE, or SMTP_RAW_MESSAGE, you must add the files that you specify as content files in the Content Files Tab before running the test.


To add content files:

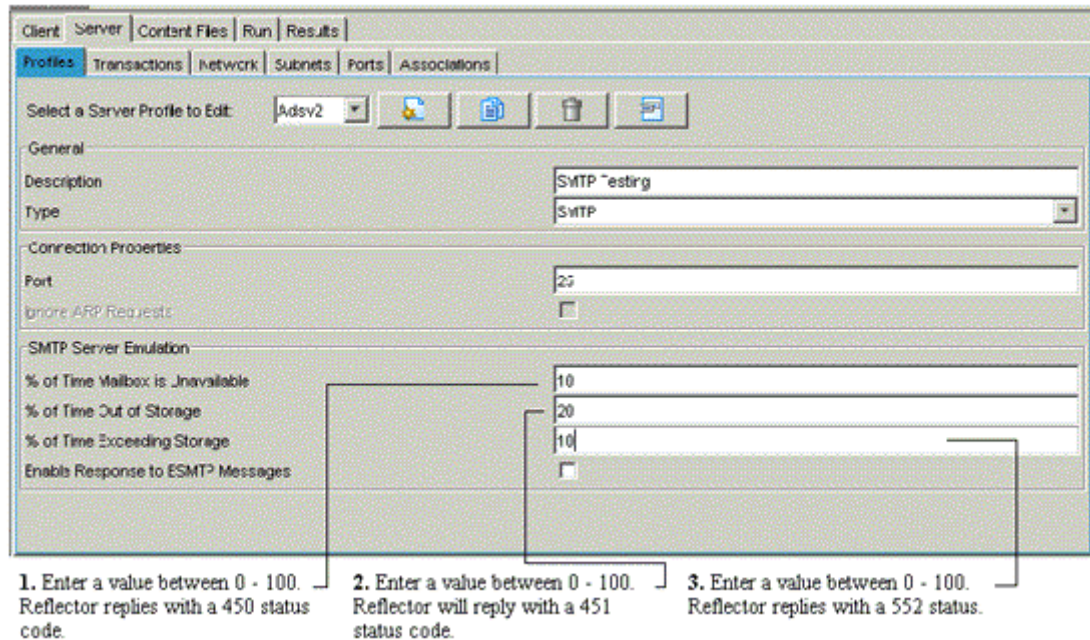
1. Click the **Content Files** tab.
2. Click the **Add Content File** button . A directory dialog box appears. By default, it lists files that appear in the default content file directory. When you first install the Avalanche software, this directory contains sample streaming files for use with a streaming test. You can also store your own content files in this directory.
3. Select the file, and then click the **Add** button. The file appears in the **Content File** tab.

Defining the Server Profile

Complete this section only if you are defining a Device test.

To define an SMTP server profile:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the test, and then select **SMTP** as the server type.
4. Enter the port where the server resides. By default, SMTP uses port 25 for transport.
5. Set SMTP server emulation settings. By default they are 0.



Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test** icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

NOTE: The statistical data for SMTP tests are described in SMTP Statistics in the Help.

Testing Telnet

To complete Telnet-specific information and run the test:

1. Learn about Avalanche Telnet testing.
2. Define Telnet commands on the Client Profile tab.
3. Define Telnet commands on the Server Profile tab. (Device test only)
4. Create a forms database. (If using a forms database to supply client data.)
5. Add an Actions list that defines which profile (and forms database if applicable) to use.
6. Run the test and review results.

About Telnet Testing

Avalanche can emulate a Telnet client and send commands to Reflector or a Telnet server. You specify the commands to be sent in the Telnet Commands pane on the Client Profile tab.

If you run the test against Reflector, use the Server Profile tab to set up Reflector to expect and respond to the commands from Avalanche. If you run the test against a real Telnet server, set up the Avalanche Telnet client profile with the exact response the Telnet server provides to the command.


After defining your profiles, create an Actions list to tell the test which profile to use to obtain the Telnet commands. (You can also use an Actions list to refer to a forms database that defines Telnet commands.)

NOTE: Telnet is a send and expect protocol designed to work in conjunction with Avalanche. You must configure the flow of the session in both Reflector (or a server) and Avalanche consistently. That which is sent by Avalanche is expected by Reflector, and vice-versa.

Defining Client Telnet Commands

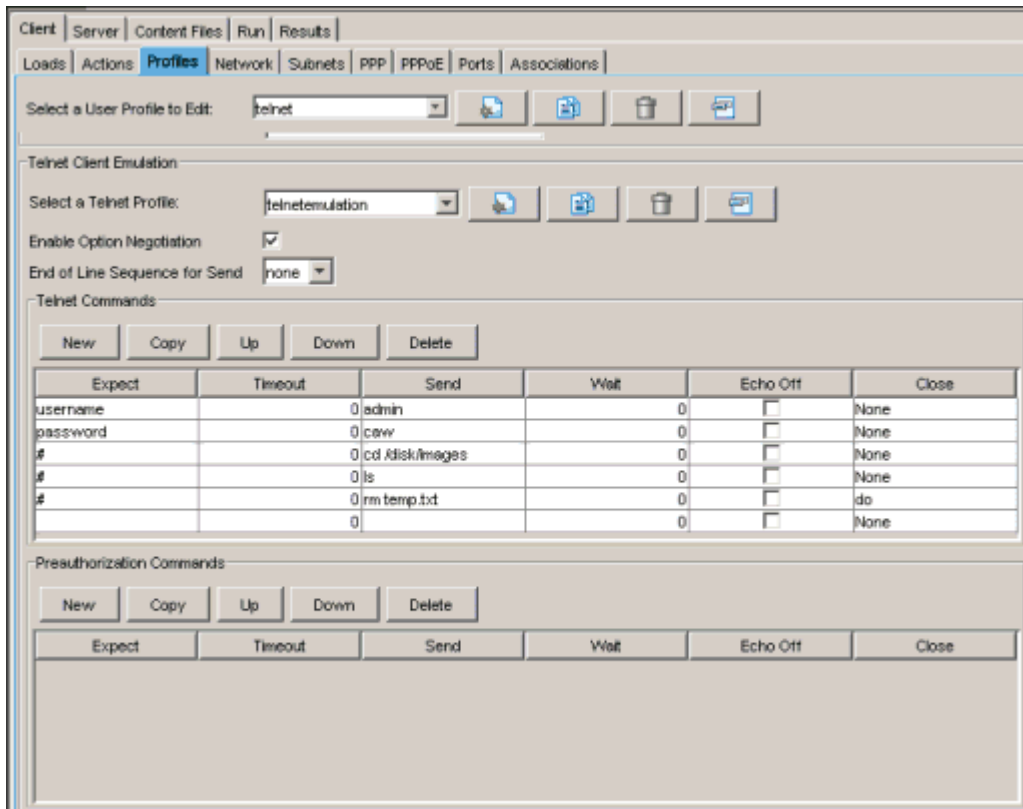
Use the Client Profile tab to define client Telnet commands.

To define a client profile:

1. Click the **Client** tab, and then the **Profiles** tab.
2. Select the client profile with which you want to work.
3. In the Telnet Client Emulation pane, click the **New** button  to create a new Telnet profile.
4. Select if you want to enable option negotiation or send end of line sequences for a send.
5. Click the **New** or **Copy** Telnet Command buttons to add or copy a command.

6. Within the Telnet table, define the commands and values that you want to use. These should be the same sequence of commands that you would use if performing a live Telnet session. For example, you might expect a username prompt, and then send a username; expect a password prompt, and then send a password; expect a command prompt, and then send a command such as ls, and so on. You can enter specific values or reference a column of data in a forms database for values. (For information about forms databases, see ["Creating a Forms Database for a Telnet Test."](#))
7. Use the Preauthorization Command buttons to add or copy a command. Define Preauthorization commands only for a network topology where the Telnet session will go through a double-authentication procedure. Within the Preauthorization table, define the commands that you want to use.


The following shows an example where Telnet commands are defined on the Client Profiles tab.



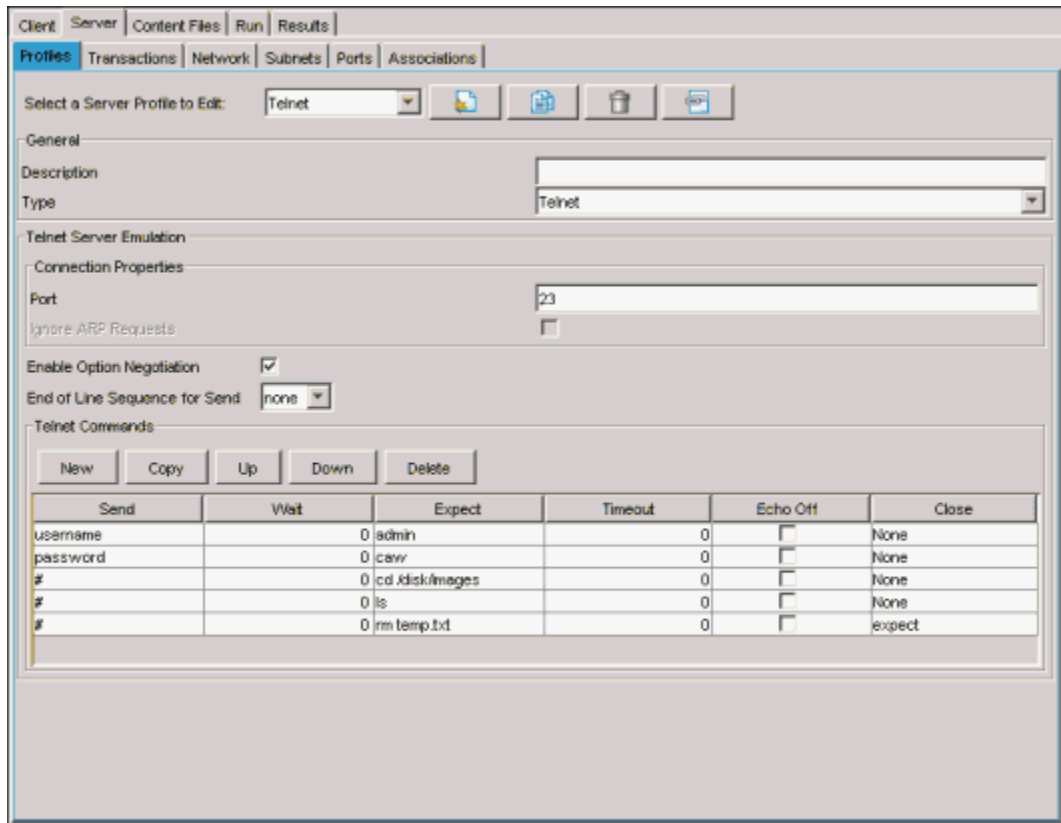
Defining Reflector Telnet Commands

Complete this section only if you are defining a Device test. Use the Server Profile tab to define Telnet commands that Reflector should expect to receive from the client and send to the client.

To define a server profile:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the profile, and then select **Telnet** as the server type.
4. Enter the port where the server resides, and select Telnet server emulation options.
5. In the Telnet Commands pane, click the **New** or **Copy** buttons to add or copy a command.
6. Within the Telnet table, define the commands and values that you want to use. The commands that you enter should correspond exactly to the commands that you entered for the client. For example, if the client expects username, the first command for reflector is to send username.

The following shows an example.



Send	Wait	Expect	Timeout	Echo Off	Close
username		admin	0	<input type="checkbox"/>	None
password		cavv	0	<input type="checkbox"/>	None
#		cd /disk/images	0	<input type="checkbox"/>	None
#		ls	0	<input type="checkbox"/>	None
#		rm temp.txt	0	<input type="checkbox"/>	expect

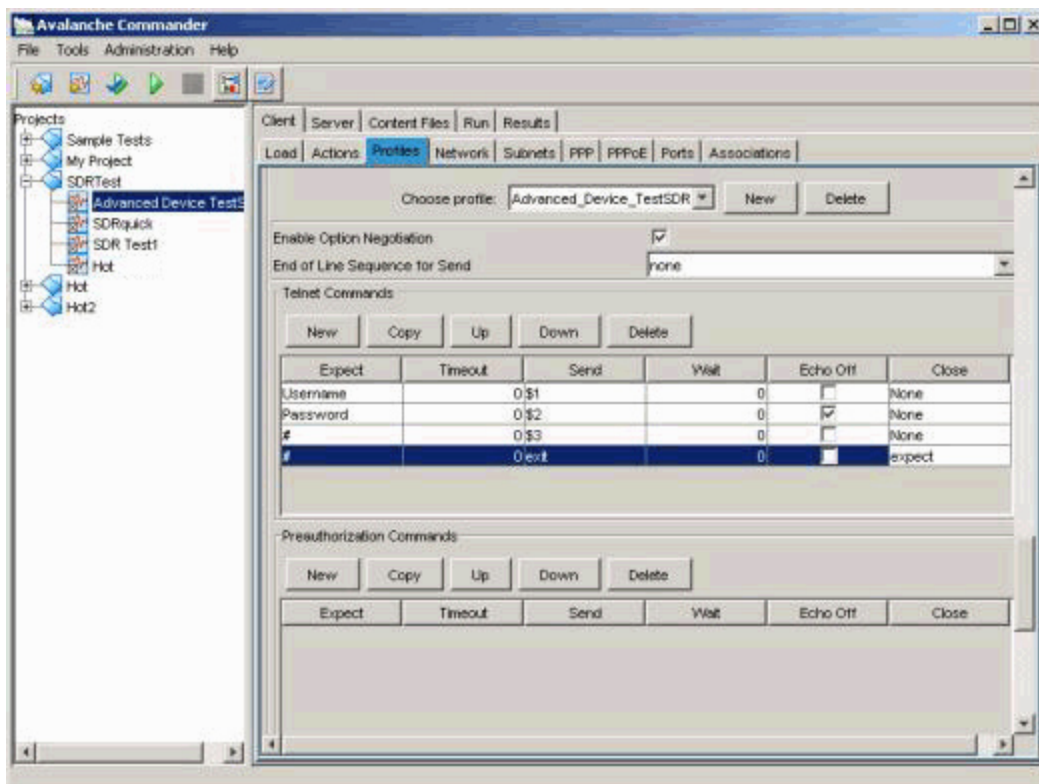
NOTE: The close command that you specify at the end of the session should be **do** on the client side, and **expect** on the server side, or vice-versa.

Creating a Forms Database for a Telnet Test

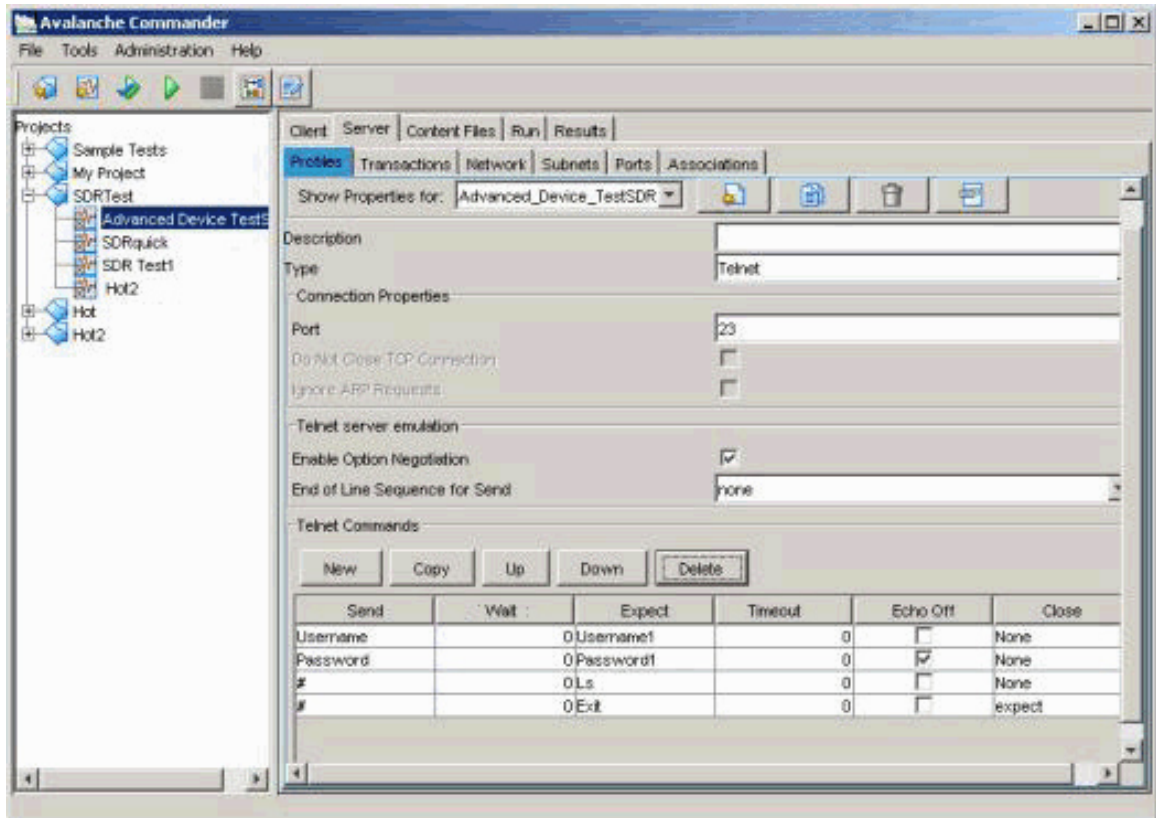
NOTE: Complete this section only if you want to use a forms database to supply Telnet information.

You can also use a forms database to supply information for the client for use with a Telnet protocol. (You cannot use a forms database to supply information for the server.) In the Telnet Commands pane, you identify the form database column of data that you want to send by entering \$ and the form database column number such as \$1 to refer to the first column, \$2 to the second, and so on. For example, in the following image, the \$1 in the first Send field refers to the first column of data which will supply a username. You identify the forms database file name in an Actions list associated with the test. (For more information about Action Lists, see Using Forms Databases with Action Lists later in this topic.)

The following image shows how the Client Profiles tab is set up to reference columns 1, 2, and 3 (\$1, \$2, \$3) of a forms database.




The following image shows how the Server Profiles tab is set up to respond to the client.



NOTE: Servers do not support a forms database. So that the server is ready to respond to the information sent from the client, create server profiles that include Send and Expect values corresponding to each column of data in the forms database and for each SimUser accessing the database. For example, in the previous image of the Server Profiles tab, Send and Expect data indicates username and password values that the server should expect.

To create a forms database:

1. Create or select the profile with which you want to work on the Client Profile tab.
2. Click the **New** button  that appears next to the **Use Form Database** field. The **Create New Form Data** window appears.
3. Complete entries in the Create New Form Data window, and then click **OK**.


The following entries in a forms database could support the examples shown previously in this section:

```
joe,joepass,ls
maria,mariapass,ls
fred,fredpass,ls
harry,harrypass,ls
jose,josepass,ls
kate,katepass,ls
thuy,thuypass,ls
lee,leepass,ls
fran,franpass,ls
rajiv,rajivpass,ls
```

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom. You can create an Action list for Application tests or for Device tests.

To create an Actions list:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.
3. Enter a description for the Actions list.
4. Define the Telnet action as follows.

Syntax

```
telnet://IP address[:port] PROFILE=profile name [FORMS=forms database name]
```

IP address:port number—IP address for the test. (Port is optional; if it is not included, then 23 is assumed.)

PROFILE=filename—Specifies the Telnet profile to use with this Action list. The profile name that you specify is the profile that you select in the Telnet Client Emulation pane from the Choose Profile drop-down menu on the Client Profiles tab.

FORMS=filename—If you use forms database values in the Telnet profile (example username = \$1), assign the name of the database here. Different Actions can use different forms databases, but only one forms database can be used on each separate Action.

Examples

An IPv4 address:

```
telnet://192.168.42.11 PROFILE=telnet_profile
```

An IPv6 address:

```
telnet://[2106::0200:FF:FE00:8102] PROFILE=telnet_profile
```

An IPv4 address with reference to a forms database:

```
telnet://192.168.42.11 PROFILE=Telnet FORMS=telnet_db
```

Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:



Click the **Run Test** icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

Testing VOD Multicast

To complete VOD Multicast-specific information and run the test:

1. Learn about Avalanche VOD Multicast testing.
2. Add an Actions list.
3. Define the server profile. (Device test only)
4. Add Content File (Spirent-RTP or MPEG-2 transport method only)
5. Run the test and review results.

About VOD Multicast Testing

VOD is deployed extensively internationally and in the United States. It is important because of the need to test Triple Play (Voice, Video, and Data). Avalanche VOD multicast support provides extensive capabilities for testing the video part of triple play deployments. With Avalanche VOD multicasting emulation, the client emulates TV viewers who are surfing or changing channels. This test helps you assess the behavior of VOD infrastructures. To exercise and measure Digital Subscriber Line Access Multiplexer (DSLAM)


behavior, the client emulates TV viewers, while the server emulates video streaming servers.

Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom.

Each line in the Action list represents a channel (multicast group address) to which to tune. You can also intermix any of the supported Action schemes. For example, interspersing `http://` emulates a user accessing the Internet, between tuning to streaming channels.

To add an Actions list for a VOD Multicast test:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.

Enter information that emulates multicasting. Refer to the following syntax and examples.

Syntax

```
time_in_msMCAST://mcast_group_addr
```

or

```
time_in_msMCRTP://mcast_group_addr
```

`Time_in_ms`—Specifies the time, in milliseconds, to remain attached to the channel.

`MCAST://`—Specifies raw UDP. This must match the transport method that you choose in the Server Profile tab.

`MCRTP://`—Specifies RTP. This must match the transport method that you choose in the Server Profile tab.

`Mcast_group_addr`—Specifies the multicast group address. This must match the group address that you choose in the Server Profile tab.

NOTE: If you use RTP streams (`mcrtp://` channels listed in the Action list), statistics related to RTP streaming are counted. The `mcrtp://` scheme is valid only if the server is producing RTP data on the channel. If the server is broadcasting some other data format, then the RTP statistics will be meaningless.

Example

In the following example, the viewer tunes to each of .1, .2, and .3 channels for five seconds, then settles on .4 for an hour:

```
5000 MCRTP://225.1.1.1
5000 MCRTP://225.1.1.2
5000 MCRTP://225.1.1.3
3600000 MCRTP://225.1.1.4
```

In the following user session example, the viewer tunes to each of .1, .2 and .3 channels for five seconds, then settles on .4 for an hour:

```
5000 MCRTP://225.1.1.1
5000 MCRTP://225.1.1.2
5000 MCRTP://225.1.1.3
3600000 MCRTP://225.1.1.4
```


The previous examples use an IPv4 address. The following example uses an IPv6 address:

```
5000 MCAST://[FF0E::2222:2222:1]
```

Defining the Server Profile

Complete this section only if you are defining a Device test.

To define a VOD Multicast server profile:

1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the test, and then select **Mcast** as the server type.
4. Enter the port where the server resides. The default port is 2001.
5. Set server emulation settings.

The screenshot shows a configuration window titled "Multicast server emulation" with the following fields and options:

- Group Address: 225.1.1.1
- Group Address Increment: 1
- Transport Method: raw UDP rtp
- Content Type: Generate dynamically Spirent-RTP MPEG-2
- Interpacket Delay (ms): 100
- Data Size: 1024
- Content: s

1. Enter the VOD multicast group IP address onto which you want this server to stream data. This is the **destination** IP address of packets sent by this server.

2. If you specified a range of addresses on the Associations tab, then the Group Address Increment specifies the incremental value of the multicast group address onto which each server streams data.

3. Select a transport method and content type. You cannot use MPEG-2 if the transport method is rtp.

4. If you select Generate dynamically as the content type, enter the time, in milliseconds, between sending each packet (default is 100), and enter the size of the packet in bytes.

If you select Spirent-RTP or MPEG-2, enter the name of the streaming file that you will add as a content file.


NOTE: Specify the **source** address on the Server Associations tab and by entering a Server IP Address Range on the Subnet to which this server is attached.

Adding Content Files

If you select Spirent-RTP or MPEG-2 as the transport method, add the file that you identified on the server profile.

NOTE: Spirent-RTP is a Spirent-proprietary format that consists of pre-formed RTP packets. Spirent supplies several files of this format.


To add content files:

1. Click the **Content Files** tab.
2. Click the **Add Content File** button . A directory dialog box appears. By default, it lists files that appear in the default content file directory.
3. Locate and select the file that you want, and then click the **Add** button. The file appears in the **Content File** tab.

Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.

Testing Windows Media (MMS)

To complete Windows Media-specific information and run the test:

1. Learn about Avalanche Windows Media testing.
2. Add an Actions list
3. Add content files (Device test only)
4. Define the server profile (Device test only)
5. Run the test and review results

About Windows Media Testing

With version 9, Windows Media has gained widespread acceptance in the streaming community and is poised to achieve even more penetration in the future. Avalanche emulates a number of Microsoft Windows Media Players that retrieve Windows media files. Avalanche supports MMS (Microsoft's proprietary streaming protocol) and RTSP/RTP over the following transport protocols: udp, tcp and http. (See Testing RTSP/RTP Streaming for more information about RTSP/RTP.)


Adding an Actions List

Action lists identify the actions that Avalanche takes for each simulated user during a test. Each list line represents a simulated user requesting an object from your test server. As Avalanche simulates new users, each one uses the list from top to bottom. An Action list uses two elements: Level and URI. Enter a 1 preceding the URI to signify a top-level retrieval. Within your Actions list, you identify media files.

For each Action list entry, Avalanche establishes a connection, requests playback, and then terminates the connection when it has completed.

NOTE: The Ramp Down phase time in the load profile must have enough time to allow all the streams to finish playing; otherwise, uncompleted sessions are counted as failures. See the Loads Tab in the help for more information.

To add an Actions list for a Windows-Media test:

1. Click the **Client** tab, and then the **Actions** tab.
2. Click the **New** button  to create a new list.

Enter information that simulates retrieving media. Refer to the following syntax and example.

Syntax

```
1 mms://server_IP_address/directory/filename.asf  
1 mms://server_IP_address/filename.asf
```

Server IP address—The address of the server, such as 10.1.79.1.

Directory-The directory where the file is stored. Define this information if you are testing against an actual server. It is not required if you are using Reflector or a SMartBits module.

Filename-The name of the media file.

NOTE: .asf is the only MMS file format supported.


Example

```
1 mms://192.168.42.11/mymmsfiles/welcome1.asf
1 mms://192.168.42.11/welcome1.asf
```

Adding Content Files

Add the specified media file as a content file before you start the test.


To add content files:

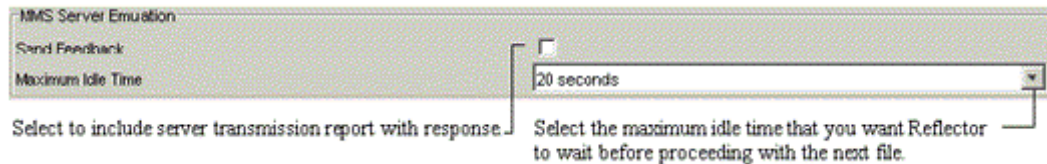
1. Click the **Content Files** tab.
2. Click the **Add Content File** button . A directory dialog box appears. By default, it lists files that appear in the default content file directory. (When you first install the Avalanche software, this directory contains sample streaming files.) Store your content files in this directory.
3. Select the file, and then click the **Add** button. The file appears in the **Content File** tab.

Defining the Server Profile

Use a server profile to define the port and server emulation.

To define a server profile:


1. Click the **Server** tab, and then the **Profiles** tab.
2. Click the **New** button  to create a new profile.
3. Enter a description for the test, and then select **MMS** as the server type.
4. Enter the port where the server resides. The default port is 1755.
5. Select server emulation settings.



Running the Test and Reviewing Results

After completing all set-up steps, run the test.

To run the test:

Click the **Run Test**  icon. The Monitor tab appears. From this tab you can monitor test statistics as Avalanche reports them. For example, click the **Client Stats** or **Server Stats** buttons to view real-time stats and graphical representations.

To view results:

Click the **Results** tab.