Network Working Group
Internet-Draft
Expiration Date:

Terry Martin
M2networx INC
B. Hickman
Netcom Systems
November 2000

Benchmarking Methodology for Firewalls

<draft-ietf-bmwg-firewall-01.txt>

Status of this Memo

Table of Contents

1. Introduction

This document is intended to provide methodology for the benchmarking of firewalls. It provides methodologies for benchmarking forwarding performance, connection performance, latency and filtering. In addition to defining the tests, this document also describes specific formats for reporting the results of the tests.

A previous document, "Benchmarking Terminology for Firewall Performance" [1], defines many of the terms that are used in this document. The terminology document SHOULD be consulted before attempting to make use of this document.

2. Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

3. Scope

Firewalls can provide a single point of defense between two networks--it protects one network from the other. Usually, a firewall protects the company's private network from the public or shared networks to which it is connected. A firewall can be as simple as a device that filters different packets or as complex as a group of devices providing solutions that offers combined packet filtering

and application-level proxy or network translation services. This RFC will focus on developing benchmark testing of systems from an application perspective and will be developed independent of any firewall implementation. These tests will evaluate the ability of

firewall devices to control and manage applications services used by today's businesses such as applications like the World Wide Web, File transfer procedures and e-mail.

Even through there are many different control methods of managing application level being implemented, this RFC does not condone or promote any aforementioned process or procedure. It's goal is to present a procedure that will evaluate firewall performance independent of their implementation.

4. Test Setup

Test configurations defined in this document will be confined to dual-homed and tri-homed as shown in figure 1 and figure 2 respectively.

Firewalls employing Dual-Homed configurations connect two networks. One interface of the firewall is attached to the unprotected network, typically the public network(i.e. - Internet). The other interface is connected to the protected network, typically the internal LAN. In the case of Dual-Homed configurations, servers which are made accessible to the public(Unprotected) network are attached to the private(Protected) network.

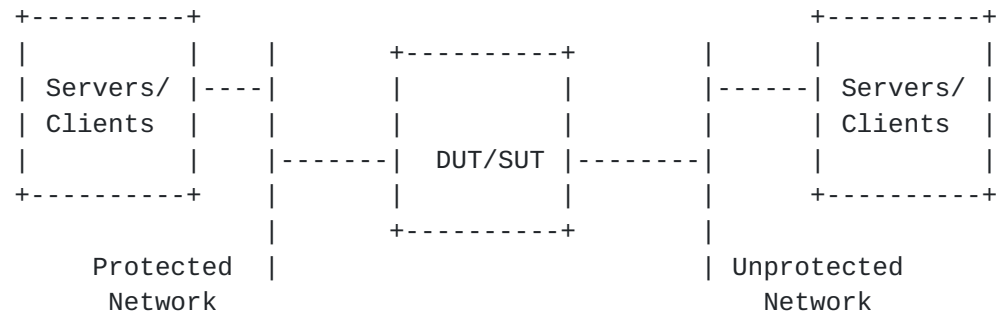

Figure 1(Dual-Homed)

Tri-homed[1] configurations employ a third segment called a DMZ. With tri-homed configurations, servers accessible to the public network are attached to the DMZ. Tri-Homed configurations offer additional security by separating server accessible to the public network from internal hosts.

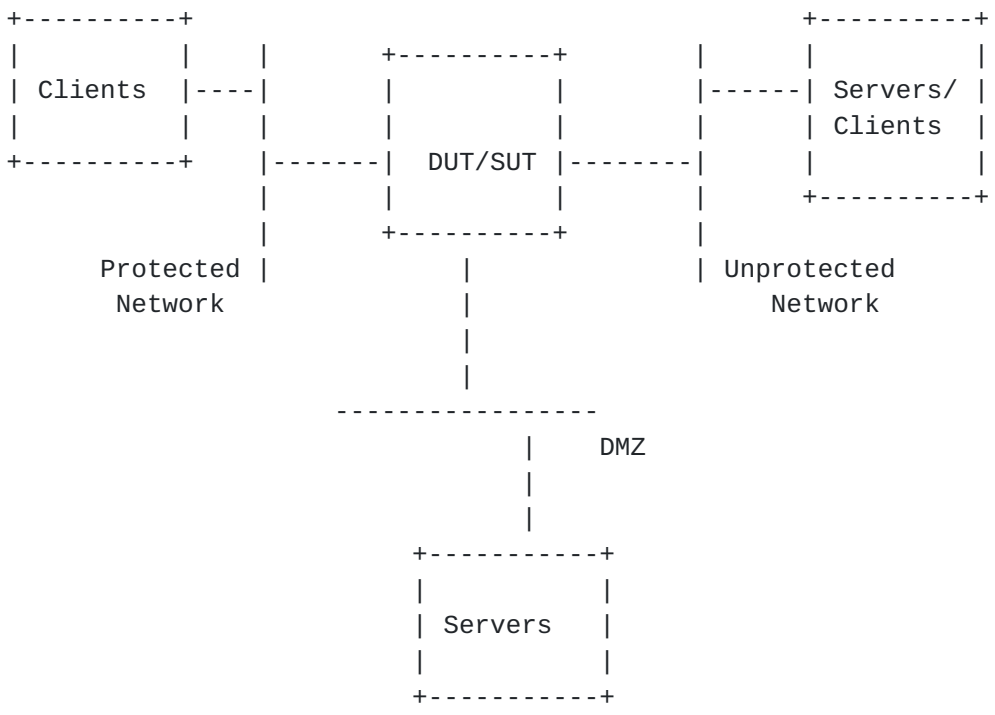

Figure 2(Tri-Homed)

4.1 Test Considerations

4.1.1 Virtual Clients/Servers

Since firewall testing may involve data sources which emulate multiple users or hosts, the methodology uses the terms virtual clients/servers. For these firewall tests, virtual clients/servers specify application layer entities which may not be associated with a unique physical interface. For example, four virtual clients may originate from the same data source[1]. The test report SHOULD indicate the number of virtual clients and virtual servers participating in the test on a per interface(See 4.1.3) basis.

Need to include paragraph for synchronize start of test. Data sources MUST be synchronized to start initiating connections within a specified time of each other.

4.1.2 Test Traffic Requirements

While the function of a firewall is to enforce access control policies, the criteria by which those policies are defined vary depending on the implementation. Firewalls may use network layer and/or, in many cases, application-layer criteria to make access-control decisions. Therefore, the test equipment used to benchmark the DUT/SUT performance MUST consist of real clients and

servers generating legitimate layer 7 conversations.

The tests defined in this document use HTTP, FTP, and SMTP sessions for benchmarking the performance of the DUT/SUT. Other layer 7

conversations are outside the scope of this document. See appendices for specific information regarding the transactions involved in establishing/tearing down connections as well as object formats for each of the aforementioned protocols.

4.1.3 DUT/SUT Traffic Flows

Since the number of interfaces are not fixed, the traffic flows will be dependent upon the configuration used in benchmarking the DUT/SUT. Note that the term "traffic flows" is associated with client-to-server requests.

For Dual-Homed configurations, there are two unique traffic flows:

```
Client       Server
  ------         ------
  Protected   -> Unprotected
  Unprotected -> Protected
```

For Tri-Homed configurations, there are three unique traffic flows:

```
Client       Server
  ------         ------
  Protected ->   Unprotected
  Protected ->   DMZ
Unprotected -> DMZ
```

4.1.4 Multiple Client/Server Testing

One or more clients may target multiple servers for a given application. Each virtual client MUST initiate requests(Connection, file transfers, etc.) in a round-robin fashion. For example, if the test consisted of six virtual clients targeting three servers, the pattern would be as follows:

Client	Target Server(In order of request)			
#1	1	2	3	1...
#2	2	3	1	2...
#3	3	1	2	3...
#4	1	2	3	1...
#5	2	3	1	2...
#6	3	1	2	3...

4.1.5 NAT(Network Address Translation)

Most firewalls come with Network Address Translation(NAT)networks built in which translates internal host IP addresses attached to the

protected network to a virtual IP address for communicating across the unprotected network(Internet). This involves additional processing on the part of the DUT/SUT and may impact on performance. Therefore,

tests SHOULD be ran with NAT disabled and NAT enabled to determine the performance differentials. The test report SHOULD indicate whether NAT was enabled or disabled.

4.1.6 Rule Sets

Rule sets[1] are a collection of access control policies that determines which packets the DUT/SUT will forward and which it will reject. The criteria by which these access control policies may be defined will vary depending on the capabilities of the DUT/SUT. The scope of this document is limited to how the rule sets should be applied when testing the DUT/SUT.

The firewall monitors the incoming traffic and checks to make sure that the traffic meets one of the defined rules before allowing it to be forwarded. It is RECOMMENDED that a rule be entered for each host(i.e. - Virtual client). Although many firewalls permit groups of IP addresses to be defined for a given rule, tests SHOULD be performed with large rule sets, which are more stressful to the DUT/SUT.

The DUT/SUT SHOULD be configured to denies access to all traffic which was not previously defined in the rule set.

4.8 Web Caching

Some firewalls include caching agents in order to reduce network load. When making a request through a caching agent, the caching agent attempts to service the response from its internal resources. The cache itself saves responses it receives, such as responses for HTTP GET requests. The report SHOULD indicate whether web caching was enabled or disabled on the DUT/SUT. The test report SHOULD indicate whether NAT was enabled or disabled.

4.9 Authentication

Access control may involve authentication processes such as user, client or session authentication. Authentication is usually performed by devices external to the firewall itself, such as an authentication servers and may add to the latency of the system. Any authentication processes MUST be included as part of connection setup process.

5. Benchmarking Tests

5.1 Concurrent Connection Capacity

5.1.1 Objective

To determine the maximum number of concurrent connections supported by the DUT/SUT, as defined in RFC2647[1]. This test will employ a step search algorithm to obtain the maximum number of concurrent FTP,HTTP or SMTP connections the DUT/SUT can maintain.

5.1.2 Setup Parameters

The following parameters MUST be defined. Each parameters is configured with the following considerations.

Connection Attempt Rate - The rate at which new connection requests are attempted. The rate SHOULD be set lower than maximum rate at which the DUT/SUT can accept new connection requests.

Connection Step count - Defines the number of additional connections attempted for each iteration of the step search algorithm.

Object/Message - Defines the number of bytes to be transferred across each connection.

5.1.3 Procedure

Each virtual client will attempt to establish connections to their target server(s) at a fixed rate in a round robin fashion. Each iteration will involve the virtual clients attempting to establish a fixed number of additional connections. This search algorithm will be repeated until either:

- One or more of the additional connection attempts fail to complete
- One or more of the previously established connections failed.

Data transfers SHOULD be performed on each connection after the given connection is established. Data transfers MUST be performed on all connections after all of the addition connection have been established.

When benchmarking with FTP, virtual clients will issue NOOP command's to validate that work can be performed across each connection. The virtual clients must receive a Command Successful reply from the target server in order to be considered a valid connection.

When benchmarking with other applications such as HTTP or SMTP, validation of the connection will be performed by initiating object/message transfers. All bytes associated with the object/message transfers MUST be received by the requesting virtual client in order to be considered a valid connection.

5.1.5 Measurements

Total number of connections that were successfully completed in a step. Test equipment MUST be able to track each connection to verify all required transaction between the virtual client and server completed successfully. This includes successful completion of both the command sequences and exchanging of any data across each of those connections.

5.1.6 Reporting Format

Maximum concurrent connections reported MUST be the aggregate number of connections completed for the last successful iteration. Report SHOULD also include:

- Connection Attempt Rate.
- Connection Step Count.

A log file MAY be generated which includes for each step iteration:

- Pass/Fail Status.
- Total connections established.
- Number of previously established connections dropped.
- Number of the additional connections that failed to complete.

5.2 Maximum Connection Setup Rate

5.2.1 Objective

To determine the maximum connection rate which can be supported through the DUT/SUT. As with the Concurrent Connection Capacity test, FTP,HTTP and SMTP sessions will be used to determine this metric.

5.2.2 Setup Parameters

The following parameters MUST be defined. Each test parameter is configured with the following considerations.

Initial Attempt Rate - The rate at which the initial connection requests are attempted.

Number of Connections - Defines the number of connections that must be established. The number MUST be between the number of participating virtual clients and the maximum number supported by the DUT/SUT. It is RECOMMENDED not to exceed the concurrent connection capacity found in section 5.1. The connection rate may vary depending on the number of connections attempted.

Object/Message - Defines the number of bytes to be transferred across each connection.

5.2.3 Procedure

An iterative search algorithm will be used to determine the maximum connection rate. This test iterates through different connection rates with a fixed number of connections attempted by the virtual clients to their associated server(s).

Each iteration will use the same connection establishment and connection validation algorithms defined in the concurrent capacity test(See 5.1). After each iteration, the tester MUST close all connections prior to continuing to the next iteration.

5.2.4 Measurements

The highest connection rate, in connections per second, for which all connections completed successfully. Test equipment MUST be able to track each connection to verify all required transaction between the virtual client and server completed successfully. This includes successful completion of both the command sequences and exchanging of any data across each of those connections.

5.2.5 Reporting Format

The maximum connection rate reported MUST be the maximum rate for which all connections successfully completed.

A log file MAY be generated which includes for each step iteration:

- Pass/Fail Status.
- Connection attempt rate.
- Number of the connections that failed to complete.
- Total connections established.

5.3 Connection Establishment Time

5.3.1 Objective

To characterize the connection establishment time[1] through or with the DUT/SUT as a function of the number of open connections.

5.3.2 Setup Parameters

The following parameters MUST be defined. Each parameters is configured with the following considerations.

Connection Attempt Rate - The rate at which new connection requests are attempted. The rate SHOULD be set lower than maximum rate at which the DUT/SUT can accept new connection requests.

Connection Step count - Defines the number of additional connections attempted for each iteration of the step algorithm.

Object/Message - Defines the number of bytes to be transferred across each connection.

5.3.3 Procedure

The test will use the same algorithm as defined in the Concurrent Capacity Test. This includes both the connection establishment and validation of each connection by transferring data across each connection.

5.3.4 Measurement

For each iteration, the tester MUST measure the Min/Avg/Max connection times for the additional connections. It is RECOMMENDED that in addition to the application layer, the tester include measurements at the lower layer protocols(i.e. - TCP, ATM) when applicable. For each of the protocols which the tester is measuring, the connection establishment time shall consist of all transactions required to enable data to be transferred across the given connection.

For example, FTP requires the user to login prior to being able to get files, view directories and so forth. Connection establishment times MUST include all of these transactions. In the case of TCP, the connection establishment time would consist of the three-way handshake between the two hosts(See Appendix D).

5.3.5 Reporting Format

Graph of the min/avg/maximum connection establishment times versus the number of open connections. The report MUST identify the layer for which

the measurement was taken(i.e. - Application, transport, etc).

5.4 Denial Of Service Handling

5.4.1 Objective

To determine the effect of a denial of service attack on connection establishment rates through the DUT/SUT. The Denial Of Service Handling test should be ran after obtaining baseline measurements from section 5.2.

When a normal TCP connection starts, a destination host receives a SYN (synchronize/start)packet from a source host and sends back a SYN ACK (synchronize acknowledge). The destination host must then hear an ACK (acknowledge) of the SYN ACK before the connection is established. The TCP SYN attack exploits this design by having an attacking source host generate TCP SYN packets with random source addresses towards a victim host, thereby consuming that hosts resources.

Some firewalls employ one or more mechanisms to guard against SYN attacks. If such mechanisms exist on the DUT/SUT, tests SHOULD be ran with these mechanisms enabled to determine how well the DUT/SUT can maintain the baseline connection rates determined in section 5.2 under such attacks.

5.4.2 Setup Parameters

The following parameters MUST be defined. Each parameter is configured with the following considerations.

Initial Attempt Rate - The rate at which the initial connection requests are attempted.

Number of Connections - Defines the number of connections that must be established. The number MUST be between the number of participating clients and the maximum number supported by the DUT/SUT. It is RECOMMENDED not to exceed the concurrent connection capacity found in section 5.1.

SYN Attack Rate - Defines the rate at which the server(s) are targeted with TCP SYN packets.

5.4.3 Procedure

This test uses the same procedure as defined in the maximum connection setup rate, with the addition of TCP SYN packets targeting the server(s) IP address or NAT proxy address.

The tester originating the TCP SYN attack MUST be attached to the Unprotected network. In addition, the tester MUST not respond to the SYN ACK packets sent by target server in response to the SYN packet.

5.4.4 Measurements

The highest connection rate, in connections per second, for which all legitimate connections completed successfully. Test equipment MUST be able to track each connection to verify all required transaction between the virtual client and server completed successfully. This includes successful completion of both the command sequences and exchanging of any data across each of those connections.

In addition, the tester SHOULD track SYN packets associated with the SYN attack which the DUT/SUT forwards on the protected or DMZ interface(s).

5.4.5 Reporting Format

The maximum connection rate reported MUST be the maximum rate for which all connections successfully completed. The report SHOULD include what percentage of TCP SYN packets were forwarded by the DUT/SUT.

A log file MAY be generated which includes for each step iteration:

- Pass/Fail Status.
- Connection attempt rate.
- Number of the connections that failed to complete.
- Total connections established.

5.5 Single Application Goodput

This section defined the procedures for base lining the Goodput[1] of the DUT/SUT for FTP, HTTP and SMTP traffic.

5.5.1 FTP Goodput

5.5.1.1 Objective

The File Transfer Protocol is a common application used by companies to transfer files from one device to another. Evaluating FTP Goodput will allow individuals to measure how much successful traffic has been forwarded by the DUT/SUT.

5.5.1.2 Setup Parameters

The following parameters MUST be defined. Each parameter is configured with the following considerations.

Number of Connections - Defines the number of connections to be opened for transferring data objects. Number MUST be equal or greater than the number of virtual clients participating in the test. The number SHOULD be a multiple of the virtual client

participating in the test.

Connection Rate - Defines the rate at which connections are established.

Object Size - Defines the number of bytes to be transferred across each connection.

5.5.1.3 Procedure

Each virtual client will establish a FTP connection to its respective server(s) in a round robin fashion at the connection rate. The transaction involved in establishing the FTP connection should follow the procedure defined in Appendix A.

After the login process has been completed, the virtual client will initiate a file transfer by issuing a "Get" command. The "Get" command will target a predefined file of Object Size bytes.

Once the file transfer has completed, the virtual client will close the FTP connection by issuing the QUIT command.

5.5.1.4 Measurement

The Goodput for each interface of the DUT/SUT MUST be measured. See appendix D for details in regards to measuring the Goodput of the DUT/SUT. Only file transfers which have been completed are to be included in the Goodput measurements.

The average transaction time each object successfully transferred MAY be measured. The start time will begin when the time the "Get" commands is initiated and end at the time when the client receives an acknowledgment from the server that file transfer has completed.

5.5.1.5 Reporting Format

The Goodput for each interface of the DUT/SUT MUST be reported in bits per second. This will be the aggregate of session Goodput's measured for a given interface.

Failure analysis:

The report SHOULD include the percentage of connections that failed. This includes:

- Connections which failed to establish
- Connections which failed to complete the object transfer

Transaction Processing analysis:

The report SHOULD include average transaction time in transactions per second.

The report SHOULD also include the object size(Bytes) being transferred.

5.5.2 SMTP Goodput

5.5.2.1 Objective

Another application commonly in use today is the mail transfer. One the common transport mechanisms for mail messages is the Simple Mail Transfer Protocol(SMTP). The SMTP Goodput will allow individuals to measure how much successful SMTP traffic has been forwarded by the DUT/SUT.

5.5.2.2 Setup Parameters

The following parameters MUST be defined. Each parameter is configured with the following considerations.

Number of Connections - Defines the number of connections to be opened for transferring data objects. Number MUST be equal or greater than the number of virtual clients participating in the test. The number SHOULD be a multiple of the virtual client participating in the test.

Connection Rate - Defines the rate at which connections are attempted.

Message Size - Defines the number of bytes to be transferred across each connection.

5.5.2.3 Procedure

Each virtual client will establish a SMTP connection to its respective server(s) in a round robin fashion at the connection rate. The transaction involved in establishing the SMTP connection should follow the procedure defined in Appendix B.

After the greeting exchanges have been completed, the client will initiate the transfer of the message by initiating the MAIL command. The client will then send the predefined message of Object Size.

Once the message has been acknowledged as being received by the server, the virtual client will then close the connection.

5.5.2.4 Measurement

The Goodput for each interface of the DUT/SUT MUST be measured. See appendix D for details in regards to measuring the Goodput of the DUT/SUT. Only message transfers which have been completed are to be included in the Goodput measurements.

The average transaction time for each message transferred MAY be measured. The start time will begin when the time the "MAIL" command

is initiated and end at the time when the client receives an acknowledgment from the server that the message has been received.

5.5.2.5 Reporting Format

Goodput analysis:

The Goodput for each interface of the DUT/SUT MUST be reported in bits per second. This will be the aggregate of session Goodput's measured for a given interface.

Failure analysis:

The report SHOULD include the percentage of connections that failed. This includes:

- Connections which failed to establish
- Connections which failed to complete the object transfer

Transaction Processing analysis:

The report SHOULD include average transaction time in transactions per second.

The report SHOULD also include the object size(Bytes) being transferred.

5.5.3 HTTP Goodput Goodput

5.5.3.1 Objective

Another common application is the World Wide Web (WWW) application that can access documents over the Internet. This application uses the Hypertext Transfer Control Protocol (HTTP) to copy information from one system to another.

HTTP Goodput measurement is actually determined by evaluating the Forwarding rate of packets. This is based on measuring only data that has successfully been forwarded to the destination interface.

When benchmarking the performance of the DUT/SUT, consideration of the HTTP version being used must be taken into account. Appendix C of this document discusses enhancements to the HTTP protocol which may impact performance results.

5.5.3.2 Setup Parameters

The following parameters MUST be defined. Each variable is configured with the following considerations.

Number of Connections - Defines the number of HTTP connections to be opened for transferring data objects. Number MUST be equal or greater than the number of virtual clients participating in the test. The number SHOULD be a multiple of the virtual client

participating in the test.

Connection Rate - Defines the rate at which connections are attempted.

Object Size - Defines the number of bytes to be transferred across each connection.

5.5.3.3 HTTP Procedure

For the HTTP Goodput tests, it is RECOMMENDED to determine which version of HTTP the DUT/SUT has implemented and use the same version for the test. To determine the version of HTTP, the user documentation of the DUT/SUT SHOULD be consulted.

Each client will attempt to establish HTTP connection's to their respective servers a user defined rate. The clients will attach to the servers using either the servers IP address or NAT proxy address.

After the client has established the connection with the server, the client will initiate GET command(s) to retrieve predefined web page(s).

When employing HTTP/1.0 in benchmarking the performance of the DUT/SUT, only one object will be retrieved for each of the defined object sizes. After the object has been transferred, the connection should then be torn down. When defining multiple objects, object transfers must be completed and the connections closed for all of the participating clients prior testing the next object size. This process is repeated until all of the defined objects are tested.

When employing HTTP/1.1, all objects defined by the user will be requested with a GET request over the same connection. The connection should then be torn down after all of the objects have been transferred.

5.5.3.4 Measurement

The Goodput for each of the objects sizes transferred MUST be measured. See appendix D for details in regards to measuring the Goodput of the DUT/SUT. Only objects which have been successfully acknowledged by the server are to be included in the Goodput measurements.

The transaction times for each object transferred MUST measured. The transaction connection time starts when the connection is made and will end when the web pages is completely mapped on the virtual client (when the client sends an acknowledgment packet is sent from the client).

5.5.3.5 Reporting Format

Goodput analysis:

The Goodput for each interface of the DUT/SUT MUST be reported in bits per second. This will be the aggregate of session Goodput's measured for a given interface.

Failure analysis:

The report SHOULD include the percentage of connections that failed. This includes:

- Connections which failed to establish
- Connections which failed to complete the object transfer

Transaction Processing analysis:

The report SHOULD include average transaction time in transactions per second.

The report SHOULD also include the object size(Bytes) being transferred.

Version Information

Report MUST include the version of HTTP used for the test. In addition, if the HTTP/1.1 is used, the number of concurrent GET's allowable(Pipelining) SHOULD be reported.

5.6 Concurrent Application Goodput

5.6.1 Objective

To determine the Goodput of the DUT/SUT when offering a mix of FTP, SMTP and HTTP traffic flows. Real world traffic does not consist of a single protocol, but a mix of different applications. This test will allow an individual to determine how well the DUT/SUT handles a mix of applications by comparing the results to the individual baseline measurements.

5.6.2 Setup Parameters

The following parameters MUST be defined. Each variable is configured with the following considerations.

Number of Connections - Defines the aggregate number of connections to be opened for transferring data/message objects. Number MUST be equal to or greater than the number of virtual clients participating in the test. The number SHOULD be a multiple of the virtual client participating in the test.

Connection Rate - Defines the rate at which connections attempts are opened. Number MUST be evenly divided among all of the virtual clients participating in the test.

Object/Message Size - Defines the number of bytes to be transferred across each connection. RECOMMENDED message sizes still needs to be determined.

At a minimum, at least one of the following parameters MUST be defined. In addition, the cumulative percentage all the defined percentages MUST equal 100%.

FTP Percentage - Defines the percentage of traffic connections which are to consist of FTP file transfers.

SMTP Percentage - Defines the percentage of traffic connections which are to consist of SMTP Message transfers.

HTTP Percentage - Defines the percentage of traffic connections which are to consist of HTTP GET requests.

5.6.3 Procedure

This test will run each of the single application Goodput tests, for which there is a defined percentage, concurrently. For each of the defined traffic types, the connection establishment, data/message transfer and teardown procedures will be the same as defined in the individual tests.

5.6.4 Measurements

As with the individual tests, the Goodput for each of the defined traffic types MUST be measured. See appendix D for details in regards to measuring the Goodput of the DUT/SUT. Only messages/data which have been successfully acknowledged as being transferred are to be included in the Goodput measurements.

The transaction times for each of the defined applications MUST be measured. See the appropriate single application Goodput test for the specifics of measuring the transaction times for each of the defined traffic types.

5.6.5 Reporting Format

Goodput analysis:

Reporting SHOULD include a graph of the number of connections versus the measured Goodput in Mbps for each of the defined traffic types(FTP, SMTP, HTTP).

Failure analysis:

Reporting should include a graph of number of connections versus percent success for each of the defined traffic types.

Transaction Processing analysis:
Reporting should include a graph of number of virtual connections versus average transaction for each of the defined traffic types.

5.7 Illegal Traffic Handling

To determine the behavior of the DUT/SUT when presented with a combination of both legal and Illegal traffic.

5.7.1 Procedure

Still Needs to be determined

5.7.2 Measurements

Still Needs to be determined

5.7.3 Reporting Format

Still Needs to be determined

5.8 Latency

Determine the latency of application layer data through the DUT/SUT.

5.8.1 Procedure

Still Needs to be determined

5.8.2 Measurements

Still needs to be determined.

5.8.3 Reporting format

Still needs to be determined.

APPENDICES

APPENDIX A: FTP(File Transfer Protocol)

A.1 Introduction

The FTP protocol was designed to be operated by interactive end users or application programs. The communication protocol to transport this service is TCP. The core functions of this application enable users to copy files between systems, view directory listings and perform house keeping chores - such as renaming, deleting and copying files. Unlike other protocols, FTP uses two connections. One connection, called the control connection, is used to pass commands between the client and the server. The other, called the data connection, is

used to transfer the actual data(Files, directory lists, etc.).

A.2 Connection Establishment/Teardown(Control)

FTP control connections are established by issuing OPEN command targeting either the URL or a specific IP address. Since the methodology does not include DNS servers, OPEN commands should target specific IP address of target server. A TCP connection will be established between the client and target server.

The client will then begin the login process. When logging in, it is RECOMMENDED to perform the test using Anonymous FTP Login and should use the following syntax:

```
            User ID: Anonymous
            Password:  will correspond to the System ID
 (client1_1@test.net through client 1_8@test.net)
```

Once a successful login acknowledgment is received from the server, the client may then initiate a file transfer. After all transfer operations have been completed, the FTP connection may be closed by issuing a QUIT command.

A.3 Data Connection

The data connection is established each time the user requests a file transfer and torn down when the transfer is completed. FTP supports two modes of operation, namely normal mode and passive mode, which determine who initiates the data connection. In normal mode operation, the server initiates the data connection, targeting a predefined PortID specified in the PORT command. In passive mode, the client initiates the data connection, targeting the PortID returned in response to the PASV Command. It is RECOMMENDED to perform the tests in normal mode operation.

File transfers are initiated by using the "Get" or "Put" command and specifying the desired filename. The tests defined in this document will use the "Get" command to initiate file transfers from the target server to the client.

A.4 Object Format

Need to define the object format.

APPENDIX B: SMTP (Simple Mail Transfer Protocol)

B.1 Introduction

The SMTP defines a simple straight forward way to move messages between hosts. There are two roles in the SMTP protocol, one is the sender and one is the receiver. The sender acts like a client and establishes a TCP connection with the receiver which acts like a server. The transactions defined in this section will use the terms client and server in place of sender and receiver.

B.2 Connection Establishment/Teardown

Each connection involves a connection greeting between the sender(Client) and receiver(Server). The syntax used to identify each other's hostnames during this greeting exchange SHOULD be:

"SMTPRcv_<Virtual_Server>.com"
"SMTPSender_<Virtual Client>.com"

where <Virtual_Client> and <Virtual_Server> represent a unique virtual number for the client and server respectively.

The basic transactions in moving mail between two hosts involve three basic steps which are outlined below. These are:

1) Client issuing a MAIL command identifying the message originator for that session. Syntax used to identify the originator SHOULD be as follows:

connection1,2,3...@hostname

2) Client issues an RCPT command identifying the recipient of the message for that session. Syntax used to identify the recipient of the message SHOULD be as follows:

reciever1,2,3...@hostname

3) Client issues a DATA command. After receiving the acknowledgment from the server, the client will then transfer the message which MUST include a line with a period to indicate to the server the end of the message. Once the end of message is received by the server, it will acknowledge the end of message.

The client may initiate another message transfer or close the session by initiating the QUIT command.

B.3 Message Format

As Internet e-mail has evolved, SMTP extensions have been added to support both audio and video message transfers. For these firewall tests, messages SHOULD consist of plain text ASCII.

APPENDIX C: HTTP(HyperText Transfer Protocol)

C.1 Introduction

As HTTP has evolved over the years, changes to the protocol have occurred to both fix problems of previous versions as well as improve performance. The most common versions in use today are HTTP/1.0 and HTTP/1.1 and are and are discussed below.

C.2 Version Considerations

HTTP/1.1 was approved by the WWW Consortium in July 1999 as an IETF Draft Standard. This is a formal recognition of the fact that all known technical issues have been resolved in the specification which was brought out in June 1997. HTTP/1.1 is also downward compatible with HTTP/1.0. Both protocols on the popular browsers in use today. The following is a list of features that are offered in HTTP 1.1 that are not in HTTP 1.0.

- Persistent connections and pipelining

Though both use TCP for data transfer, but differ in the way it is used, with the later version being more efficient. Once a connection is opened, it is not closed until the HTML document and all objects referred by it are downloaded. This technique is called persistent connection. By serving multiple requests on the same TCP segment, many control packets (which are not part of actual data transfer) are avoided. The technique of containing multiple requests and responses within the same TCP segment over a persistent connection is called pipelining.

- Data compression

HTTP/1.1 provides for compression of documents before file transfer. Since most other objects like images and binaries are already compressed, this feature applies only to HTML and plain text documents.

- Range request and validation

Bandwidth saving measure is the introduction of two new fields in an HTTP request header, viz. If-Modified-Since: and If-Unmodified-Since:. The significance of this feature is that if a browser identifies a file in its cache, it needn't reload it unless it has

changed since the last time it was used.

- Support for multiple hosts

It is common for an ISP to host more than one Web site on a single server. In such a case, each domain requires its own IP address.

C.3 Object Format

Object SHOULD be an HTML formatted object.

Append D. GOODPUT Measurements.

The Goodput will measure the number of bits per second forwarded by the DUT/SUT and will be referenced to the application level data. The formula for determining Goodput of the DUT/SUT is as follows:

$$\text{Goodput(Bits/Sec)} = \frac{\text{ObjectSize(Bytes)} * 8}{\text{Transfer Time(Seconds)}}$$

Transfer Time starts when the first bit of the object/message is received at the destination port of the tester. The transfer time ends when the last bit of the object/message is received at the destination port of the tester.

Appendix E. References

[1] Newman, "Benchmarking Terminology for Firewall Devices", RFC 2647, February 1998.

[2] J. Postel, "Simple Mail Transfer Protocol", RFC 821, August 1982.

[3] R. Fielding, J. Gettys, J. Mogul, H Frystyk, T. Berners, "Hypertext Transfer Protocol -- HTTP/1.1", January 1997

[4] J. Postel, J. Reynolds, "File Transfer Protocol(FTP)", October 1985