J. Babiarz
Nortel Networks
K. Hedayat
Brix Networks
R. Krzanowski
Verizon
Kiho Yum
Juniper Networks

Internet Draft
Expires: May 11, 2006
November 7, 2005

A Two-way Active Measurement Protocol (TWAMP)
draft-ietf-ippm-twamp-00

Status of this Memo

Copyright Notice

Abstract

The IPPM One-way Active Measurement Protocol [OWAMP] provides a common protocol for measuring one-way metrics between network devices. OWAMP [OWAMP] can be used in both directions independently to measure one-way metrics in both directions between two network elements. However, it does not accommodate round-trip or two-way measurements. This draft proposes a Two-way Active Measurement Protocol, based on the One-way Active Measurement Protocol [OWAMP], that will accommodate two-way or round-trip measurements.

Table of Contents

1. Introduction..2
2. Terminology...3
3. Protocol Overview...3
   3.1 Relationship of Test and Control Protocols..................3
   3.2 Logical Model...3
4. TWAMP Control...5
   4.1 Connection Setup..6
   4.2 TWAMP Control Commands......................................6
   4.3 Creating Test Sessions......................................6
   4.4 Send Schedules..6
   4.5 Starting Test Sessions......................................6
   4.6 Stop-Sessions...6
   4.7 Fetch-Session...6
5. TWAMP Test..7
   5.1 Sender Behavior...7
   5.2 Reflector Behavior..8
6. Implementers Guide...12
   6.1 Complete TWAMP...12
   6.2 TWAMP Light..12
7. Security Considerations..13
8. IANA Considerations..13
9. References...13
   9.1 Normative References.......................................14

1. Introduction

The IETF IP Performance Metrics (IPPM) working group has proposed the draft standard for round-trip delay [RFC2681] metric. IPPM has also proposed a new protocol for establishment of sessions for measurement of one-way metrics [OWAMP]. Two-way Active Measurement Protocol uses the methodology and architecture of OWAMP [OWAMP] to define an open protocol for measurement of two-way or round-trip metrics. Henceforth in this document the term two-way also signifies round-trip.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [RFC2681] and indicate requirement levels for compliant implementations.

3. Protocol Overview

The Two-way Active Measurement Protocol is an open protocol for measurement of two-way metrics. It is based on OWAMP [OWAMP] and adheres to its overall architecture and design. The protocol defined in this document defines extensions and changes to OWAMP [OWAMP] as follows:

- Define a new logical entity, Session-Reflector, in place of the Session-Receiver.

- Define the Session-Reflector behavior in place of the Session-Receiver behavior of OWAMP [OWAMP].

- Define a new test packet format for packets transmitted from the Session-Reflector to Session-Sender.

- Presence of the Fetch client in the system and the support of the Fetch command by the Server are optional.

3.1 Relationship of Test and Control Protocols

Similar to OWAMP [OWAMP], TWAMP consists of two inter-related protocols: TWAMP-Control and TWAMP-Test. The relationship of these protocols is as defined in section 1.1 of OWAMP [OWAMP].

3.2 Logical Model

The role and definition of the logical entities are as defined in section 1.2 of OWAMP [OWAMP] with the following exceptions:

- Session-Receiver is called the Session-Reflector in the TWAMP architecture.

- The presence of the Fetch-Client is optional since two-way measurements do not require data retrieval from the Session-Reflector. Consequently the support for the Fetch command is optional by the Server. However, the Server may choose to implement the Fetch-Client and support the Fetch-Command to enable both one-way and two-way measurements in the same session. This is explained in more detail in section 4.7.

Several examples of possible relationship scenarios between these roles are presented below. In the first example different logical roles are played on different hosts.

```
+----------------+               +-------------------+
| Session-Sender |<-TWAMP-Test-->| Session-Reflector |
+----------------+               +-------------------+
  ^                                     ^
  |                                     |
  |                                     |
  |                                     |
  |  +----------------+<----------------+
  |  |     Server     |<-------+
  |  +----------------+        |
  |    ^                       |
  |    |                       |
  | TWAMP-Control         TWAMP-Control
  |    |                       |
  v    v                       v
+----------------+     +-----------------+
| Control-Client |     |   Fetch-Client  |
+----------------+     +-----------------+
```

Second example is similar to the first example without the Fetch-Client. In this example only two-way metrics are collected.

```
          +----------------+                +-------------------+
          | Session-Sender |<--TWAMP-Test-->| Session-Reflector |
          +----------------+                +-------------------+
            ^                                      ^
            |                                      |
            |                                      |
            |                                      |
            |  +----------------+                  |
            |  |     Server     |<-----------------+
            |  +----------------+
            |       ^
            |       |
            |   TWAMP-Control
            |       |
            v       V
          +----------------+
          | Control-Client |
          +----------------+
```

Similar to OWAMP [OWAMP] different logical roles can be played by the same host. For example, in the figure above, there could be actually two hosts: one playing the role of Control-Client, Fetch-Client, Session-Sender, and Server, and the other playing the role of Session-Reflector. This is the third example shown below.

```
          +-----------------+                  +-------------------+
          |     Server      |<------------------|                   |
          | Control-Client  |                   | Session-Reflector |
```

```
       | Session-Sender  |<--TWAMP-Test----->|                    |
       +-----------------+                    +--------------------+
```

Additionally, following the guidelines of OWAMP [OWAMP], TWAMP has been defined to allow for small test packets that would fit inside the payload of a single ATM cell (only in unauthenticated mode).

4. TWAMP Control

All TWAMP Control messages are similar in format to and follow the same guidelines defined in section 3 of OWAMP [OWAMP].

4.1 Connection Setup

Connection establishment of TWAMP follows the same procedure defined in section 3.1 of OWAMP [OWAMP].

4.2 TWAMP Control Commands

TWAMP control commands are as defined in section 3.3 of OWAMP [OWAMP] except for the optional requirement of the Fetch-Session command.

4.3 Creating Test Sessions

Test sessions creation follows the same procedure as defined in section 3.4 of OWAMP [OWAMP]. In order to distinguish the session as a two-way versus a one-way measurement session the first octet of the Request-Session command MUST be set to 5. Value of 5 indicates that this is a Request-Session for a two-way metrics measurement session.

4.4 Send Schedules

Send schedule of test packets follow the same procedure and guidelines as defined in section 3.5 of OWAMP [OWAMP].

4.5 Starting Test Sessions

Starting test sessions follow the same procedure and guidelines as defined in section 3.6 of OWAMP [OWAMP].

4.6 Stop-Sessions

Stopping test sessions follow the same procedure and guidelines as defined in section 3.7 of OWAMP [OWAMP].

4.7 Fetch-Session

The purpose of TWAMP is measurement of two-way metrics. Two-way measurements do not rely on packet level data collected by the Session-Reflector such as sequence number, timestamp, and TTL. As such the protocol does not require the retrieval of packet level data from the Server and the Fetch-Session command is optionally supported by the Server.

However, TWAMP can be used as an extension to OWAMP [OWAMP] where both one-way and two-way measurements are measured in the same session. In this case the Server MAY support the Fetch-Session command as defined in section 3.8 of OWAMP[OWAMP]. The Session-Reflector will reject the Fetch-Session request if either it does not support the Fetch-Session command or Session-Reflector cannot provide the required data. In this case the server MUST respond with a Fetch-Ack message with Accept value of 3.

5. TWAMP Test

The TWAMP test protocol is similar to the OWAMP [OWAMP] test protocol with the exception that the Session-Reflector transmits test packets to the Session-Sender in response to each test packet it receives. TWAMP defines two different test packet formats, one for packets transmitted by the Session-Sender and one for packets transmitted by the Session-Reflector. As with OWAMP [OWAMP] test protocol there are three modes: unauthenticated, authenticated, and encrypted.

5.1 Sender Behavior

The sender behavior is as defined in section 4.1 of OWAMP [OWAMP] for both packet timing and packet format. Additionally the Session-Sender records the necessary information provided by the packets transmitted by the Session-Reflector for measuring two-way metrics. The information recording based on the received packet by the Session-Sender is implementation dependent.

5.1.1 Packet Timings

Packet timings follow the same procedure and guidelines as defined in section 4.1.1 of OWAMP [OWAMP].

5.1.2 Packet Format and Content

Session-Sender packet format and content follow the same procedure and guidelines as defined in section 4.1.2 of OWAMP [OWAMP].

5.2 Reflector Behavior

When receiving packets the reflector behavior is same as Session-Receiver behavior defined in section 4.2 of OWAMP [OWAMP] with the exception of optional packet information recording. If the Session-Reflector chooses not to collect packet information for packets received from the Session-Sender, the Server will not support the Fetch-Session command. Additionally, TWAMP requires the Session-Reflector to transmit a packet to the Session-Sender in response to each packet it receives.

As packets are received the Session-Reflector will,

- Timestamp the received packet.

- In authenticated or encrypted mode, decrypt the first block (16 octets) of the packet body.

- Copy the packet sequence number into the corresponding reflected packet to the Session-Sender.

- Optionally store the packet sequence number, send time, receive time, and the TTL for IPv4 (or Hop Limit for IPv6) from the packet IP header for the results to be transferred.

- Packets not received within the Timeout are considered lost. They are optionally recorded with their true sequence number, presumed send time, receive time consisting of a string of zero bits, and TTL (or Hop Limit) of 255. The Session-Reflector will not generate a test packet to the Session-Sender for packets that are considered lost.

- Transmit a test packet to the Session-Sender in response to every received packet. The response must be generated as immediately as possible. The format and content of the test packet is defined in section 5.2.1. Prior to the transmission of the test packet Session-Reflector MUST determine the elapsed time since the reception of the packet for incorporating the value in the reflected test packet.

5.2.1 Packet Format and Content

The Session-Reflector MUST transmit a packet to the Session-Sender in response to each packet received. The Session-Reflector SHOULD transmit the packets as immediately as possible. The Session-Reflector SHOULD set the TTL in IPV4 (or Hop Limit in IPv6) in the UDP packet to 255.

The test packet will have the necessary information for calculating two-way metrics by the Session-Sender. The format of the test packet depends on the mode being used. The format of the packet is presented below.

For unauthenticated mode:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Timestamp                            |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Error Estimate        |           MBZ                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Receive Timestamp                    |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sender Sequence Number                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Sender Timestamp                         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Sender Error Estimate    |           MBZ                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                                                               .
.                         Packet Padding                        .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```


For authenticated and encrypted modes:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   |                        IZP (12 octets)                        |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          Timestamp                            |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Error Estimate        |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
   |                        IZP (6 octets)                         |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        Receive Timestamp                      |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        IZP (8 octets)                         |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
   |                        Sender Sequence Number                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        IZP (12 octets)                        |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Sender Timestamp                         |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      Sender Error Estimate    |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
   |                        IZP (6 octets)                         |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   .                                                               .
   .                         Packet Padding                        .
   .                                                               .
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Sequence Number is the sequence number of the test packet and starts with zero and is incremented by one for each subsequent packet. The generated sequence number by the Session-Reflector,

Sequence Number, is independent from the sequence number of the received packets.

Timestamp and Error Estimate are the transmit timestamp and error estimate of the test packet respectively. Sender Timestamp and Sender Error Estimate are exact copies of the timestamp and error estimate from the Session-Sender test packet that corresponds to this test packet. The format of all timestamp and error estimate fields follow the definition and formats defined by OWAMP[OWAMP].

Receive Timestamp is the time the test packet was received by the reflector. The difference between Timestamp and Receive Timestamp is the amount of time the packet was in transition in the Session-Reflector. The Error Estimate of Timestamp also applies to Receive Timestamp.

Sender Sequence Number is the Sequence Number of the packet transmitted by the Session-Sender that corresponds to this test packet.

Similar to OWAMP [OWAMP] the TWAMP packet layout is the same in authenticated and encrypted modes. The encryption operation of Session-Receiver packet follow the same rules of Session-Sender packets as defined in OWAMP [OWAMP].

The minimum data segment length is, therefore, 36 octets in unauthenticated mode, and 80 octets in both authenticated mode and encrypted modes.

The Session-Reflector TWAMP-Test packet layout is the same in authenticated and encrypted modes. The encryption operations are, however, different. The difference is that in encrypted mode both the sequence numbers and timestamps are encrypted to provide maximum data integrity protection while in authenticated mode the sequence numbers are encrypted and the timestamps are sent in clear text. Sending the timestamp in clear text in authenticated mode allows one to reduce the time between when a timestamp is obtained by a reflector and when the packet is reflected out. In encrypted mode, both the sender and reflector have to fetch the timestamp, encrypt it, and send it; in authenticated mode, the middle step is removed, potentially improving accuracy (the sequence number can be encrypted before the timestamp is fetched).

In authenticated mode, the first block (32 octets) of each packet is encrypted using AES Electronic Cookbook (ECB) mode.

Obtaining the key, encryption method, and packet padding is as defined in section 4.1.2 of OWAMP [OWAMP]. In unauthenticated mode, no encryption is applied.

6. Implementers Guide

This section serves as guidance to implementers of TWAMP. Two architectures are presented in this section for implementations where two hosts play the subsystem roles of TWAMP. Although only two architectures are presented here the protocol does not require their use. Similar to OWAMP [OWAMP] TWAMP is designed with complete flexibility to allow different architectures that suite multiple system requirements.

6.1 Complete TWAMP

In this example the roles of Control-Client, Fetch-Client, and Session-Sender are implemented in one host referred to as the controller and the roles of Server and Session-Receiver are implemented in another host referred to as the responder.

```
       controller                                responder
 +------------------+                     +--------------------+
 | Control-Client   |<--TWAMP-Control-->| Server             |
 | Fetch-Client     |                     |                    |
 | Session-Sender   |<--TWAMP-Test----->| Session-Reflector  |
 +------------------+                     +--------------------+
```

This example provides an architecture that supports the full TWAMP standard. The controller establishes the test session with the responder through the TWAMP-Control protocol. After the session is established the controller transmits test packets to the responder. The responder follows the Session-Receiver behavior of both OWAMP [OWAMP] and TWAMP as described in section 5.2. In this architecture the responder supports the Fetch-Session command. After the transmission of test packets the controller fetches the responder's information through its Fetch-Client. This architecture allows for collection of both one-way and two-way metrics.

6.2 TWAMP Light

In this example the roles of Control-Client, Server, and

Session-Sender are implemented in one host referred to as the

controller and the role of Session-Receiver is implemented in another host referred to as the responder.

```
          controller                                  responder
       +-----------------+                  +-------------------+
       |     Server      |<---------------->|                   |
       | Control-Client  |                  | Session-Reflector |
       | Session-Sender  |<--TWAMP-Test---->|                   |
       +-----------------+                  +-------------------+
```

This example provides a simple architecture for responders where their role will be to simply act as light test points in the network. The controller establishes the test session with the Server through non-standard means. After the session is established the controller transmits test packets to the responder. The responder follows the Session-Receiver behavior of TWAMP as described in section 5.2.1. The controller receives the reflected test packets and collects two-way metrics. This architecture allows for collection of two-way metrics.

This example eliminates the need for the TWAMP-Control protocol and assumes that the Session-Reflector is configured and communicates its configuration with the Server through non-standard means. Furthermore, the Server does not support the Fetch-Session command and the responder does not collect the received packet information. The Session-Reflector simply reflects the incoming packets back to the controller while copying the necessary information and generating sequence number and timestamp values per section 5.2.1.

7. Security Considerations

The security considerations of OWAMP [OWAMP] apply.

8. IANA Considerations

There are no IANA considerations associated with this specification.

9. Acknowledgements

The authors wish to thank Sharee McNab and Nick Kinraid for their comments and suggestions.

10. References

10.1 Normative References

[OWAMP]   Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., Zekauskas, M., "A One-way Active Measurement Protocol (OWAMP)", draft-ietf-ippm-owdp-11.txt, October 2004.

[RFC2681] Almes, G., Kalidindi, S., Zekauskas, M., "A Round-Trip Delay Metric for IPPM". RFC 2681, STD 1, September 1999.

Authors' Addresses

Kaynam Hedayat
Brix Networks
285 Mill Road
Chelmsford, MA 01824
US

Phone: +1 978 367 5611
EMail: khedayat@brixnet.com
URI: http://www.brixnet.com/

Roman M. Krzanowski, Ph.D.
Verizon
500 Westchester Ave.
White Plains, NY

US

Phone: +1 914 644 2395
EMail: roman.krzanowski@verizon.com
URI: http://www.verizon.com/

Kiho Yum
Juniper Networks
1194 Mathilda Ave.
Sunnyvale, CA

US

Phone: +1 408 936 2272
EMail: kyum@juniper.net
URI: http://www.juniper.com/

IPR Disclosure Acknowledgement

Disclaimer of Validity

Copyright Notice

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.