Mint 709

Operation, Administration and Maintenance & Performance Monitoring of Customer and Network Services

Ziyang Xiong

Table of Contents

Table of Figures

1. Introduction

As the technology develops, more and more service providers have emerged to provide network accessibility to people all over the world. Nowadays, the quality of network service has become the key concern for providers; however, with the dramatically increasing network scale, it has been much difficult for operators to manually monitor the network status and performance for the whole network; therefore, fault management and performance monitoring protocols are needed for network devices to cooperate with each other to monitor the whole network.

Under this situation, TWAMP and Y.1731 protocols have been proposed and well-adopted by network equipment vendors for network OAM functioning. This report is trying to cover the most important and useful details of them to bring a logical and organized overview with these protocols.

Octets within this report are represented as unsigned integers in network byte order, unless specified otherwise.

2. TWAMP

Two-Way Active Measurement Protocol is aiming to provide round-trip measuring capability in addition to OWAMP [RFC4656], which is the fundament of TWAMP.

Comparing to one-way measurement, two-way measurement has the following advantages: It doesn't require synchronization between two MEPs (Maintenance End Point) and it only requires nearly basic echo function on the remote end point.

Basically, TWAMP is comprised of two protocols: TWAMP-Control and TWAMP-Test, each corresponding to one of the two main procedures of it. TWAMP-Test is the utility used by test sessions to achieve measurements while TWAMP-Control is the tool used to manage those sessions.

2.1 Logical Model

There are several roles in this model: Session-Sender, Session-Reflector, Server and Control-Client.

Session-Sender: The starting end point of a test session, which sends the original TWAMP-Test packets and receives the corresponding responses from Session-Reflector.

Session-Reflector: The remote end point of a test session, which receives the original test packets and sends the corresponding responses back to the Session-Sender. It is worth to note that Session-Reflector does not need to collect any packet information of the test packets it receives to fulfill the requirements of the measurement.

Server: The system which communicates with Control-Clients and manages TWAMP sessions within its administrative domain.

Control-Client: The system which communicates with the server to manage test sessions and with the Session-Sender to retrieve test results.

With this model in hand, we can include the two procedures of TWAMP to make it more clear to us: TWAMP-Control protocol is used between the Control-Client and Server to facilitate the TWAMP-Control session while TWAMP-Test protocol is used between the Session-Sender and Session-Reflector to facilitate the TWAMP-Test session.

2.2 Protocol Overview

Generally, a typical TWAMP testing can be concluded with the following steps:

1. **Connection setup:** The Control-Client first sets up a TCP connection to the TWAMP port of the Server. The Server sends a greeting message to the Control-Client with the available mode options. Then the client replies the selected mode and the server responds with an acknowledgement. The connection is deemed established if the response is a positive acknowledgement.

2. **Test session setup:** After the connection is set, the Control-Client can manage test sessions with various TWAMP-Control messages. Normally, the client will send a couple of session creation messages to set up test sessions while the server needs to reply to each message respectively with an acknowledgement.

3. **Test session start and stop:** When the needed test sessions are ready, the Control-Client will send the session start message to the Server to indicate the initiation. The Server needs to respond each message with an acknowledgement. The test sessions will be activated if the response is a positive acknowledgement. The test session will stay active until a session stop message is issued by the Control-Client.

With the respect to extensibility, TWAMP inherits two main means from OWAMP. Any extension can utilize the mode options during connection setup to introduce its own design and

architecture to the protocol. Furthermore, the protocol can introduce additional commands within one specific mode to create new possibilities to the world.

2.3 TWAMP-Control

2.3.1 Connection Setup

At the beginning of the testing, the Control-Client will first set up a tcp connection with port 862 of the Server. The server should use the DSCP value of the IP header within the TCP SYN from the client for the subsequent transactions of this session. When the tcp link is up, the Server will respond with a Server-Greeting message which contains the available mode options and the required security information:

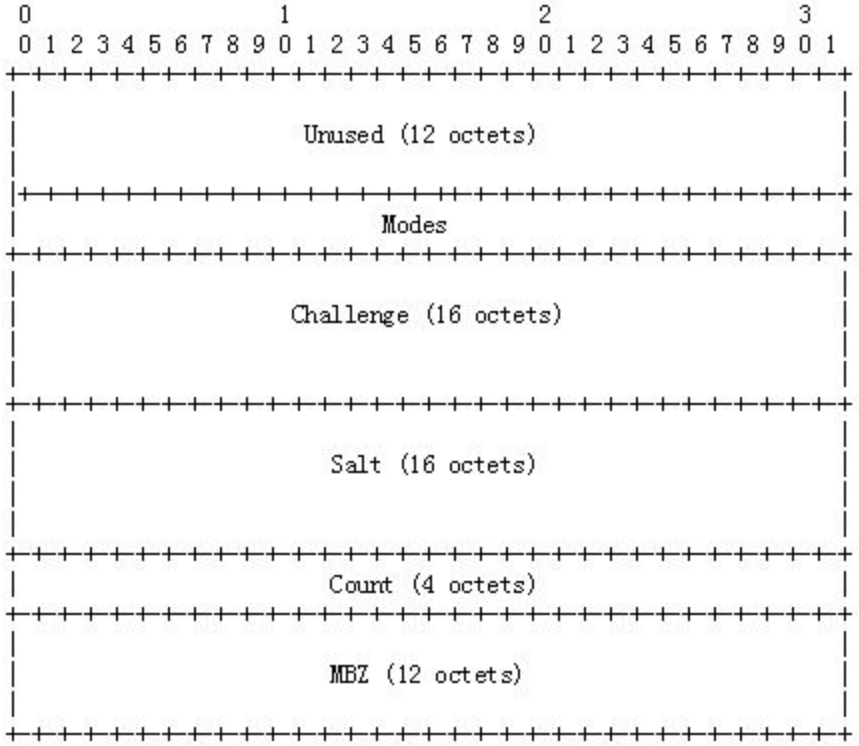

Fig 2.3.1.1 TWAMP Server-Greeting

The Modes field indicates the available mode options this server supports and all the unused bits must be set to zero. For normal TWAMP protocol, the last three bits are used and each of them indicates a specific mode.

The Challenge field is a series of bits used by the client in the latter response to prove its identity. It is generated randomly by the server.

The Salt and Count field are the information used to create the secret key for the encryption of the Token field in the client response. The salt is generated randomly by the server and the Count is a power of 2 which must be greater equal to 1024.

MBZ means must be zero. It is a common field used in most of the TWAMP messages to support extensibility. As a result, all the unused bits of this field must be set to zero by the sender and ignored by the receiver.

Upon receiving the greeting, the client needs to pick a mode from the list to run.If the client does not want to continue the test, it may close this connection with the server; otherwise, it must respond to the server with the following Set-Up-Response message indicating the desired mode and related security information:

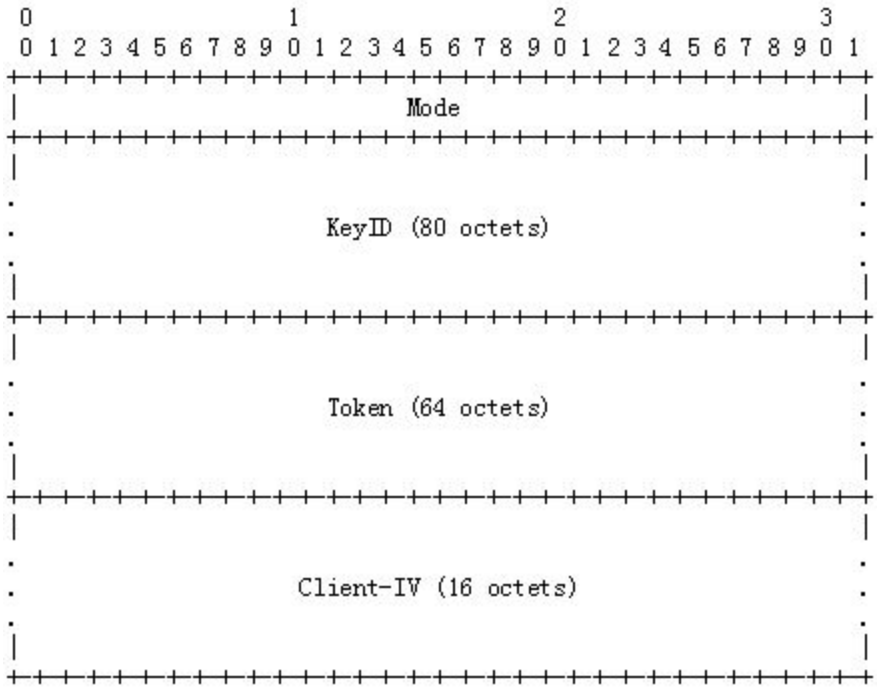

Fig 2.3.1.2 TWAMP Set-Up-Response

The Mode field here indicates the selected mode to use for the communication of this control session and all the test sessions belonging to it. It shares the same format with the Modes field in the Server-Greeting message. If zero bits are set, this indicates that the client does not want to continue the test; In this case, the server and the client should close this connection.

The KeyID contains a randomly generated UTF-8 string used to generate the secret key to protect the following Token; and zero padding will be needed if the string is shorter than 80 octets. It will be explained further in the integrity and security section.

The Token is container of the randomly generated Session-keys used for the later communication. It is a series of the 16-octet challenge from the Server-Greeting message to confirm the client's identity, a 16-octet AES Session-key and a 32-octet HMAC Session-key. It is protected by encryption with KeyID.

The Client-IV is a randomly generated 16-octet initialization vector used for the encryption of messages sent to the client.

After the receipt of the Set-Up-Response message, the server will respond with a Server-Start message which contains an acknowledgement and the required time and security information:

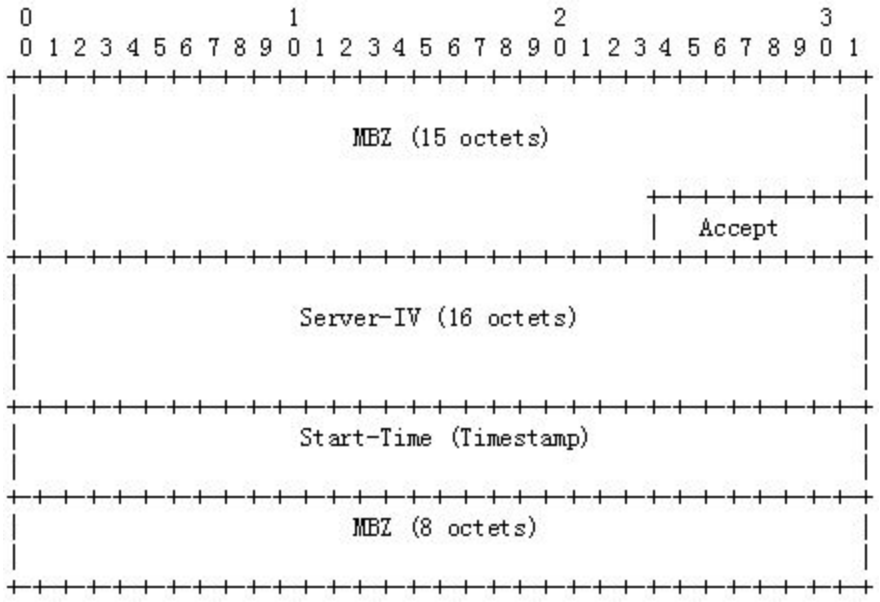

Fig 2.3.1.3 TWAMP Server-Start

Accept is an another common field used to deliver the server acknowledgement back to the client. It does not only exist within Server-Start message, but also within other acknowledgement messages the server sends back to the client. Normally, the following values are being used:

0 OK.
1 Failure, reason unspecified (catch-all).
2 Internal error.
3 Some aspect of request is not supported.
4 Cannot perform request due to permanent resource limitations.

5 Cannot perform request due to temporary resource limitations.
It is worth noting that all values other than the used ones will be deemed as 1.

The Server-IV is a randomly generated initialization vector used for the encryption of messages sent to the server.

The Start-Time indicates the start time of the server instantiation. However, it should be set to zero when the accept value is not equal to 0. The format of this field is called timestamp and it is as below::

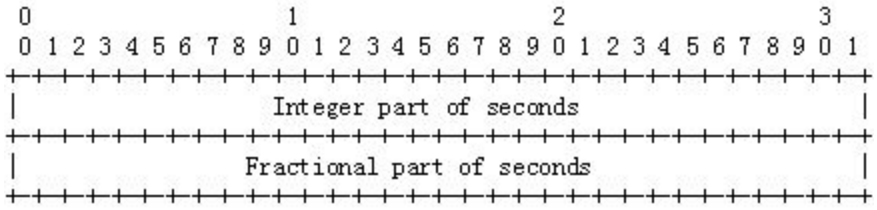

Fig 2.3.1.4 TWAMP timestamp

The Timestamp indicates a timepoint by how many seconds have passed since 0h 1st Jan 1900. As their names imply, the value is presented with two sections: integer part and fractional part.

It is worth noting that the server may close the connection if it does not accept the Set-Up-Response within this session.

After the client receives the Server-Start message, the client should close the connection if it is a negative acknowledgement. If not, the connection setup phase is deemed completed.

Several commands become available to utilize after the connection setup. Request-TW-Session and Start-Sessions commands can only be issued by the client while Stop-Sessions can be given from both parties. Since each command is represented by a message, the server will respond to each command it receives with an acknowledgement or a Stop-Sessions command.

2.3.2 Test session setup

Before testing, the client needs to create the test sessions to be used through Request-TW-Session command which is a single message containing information regarding the corresponding test session:

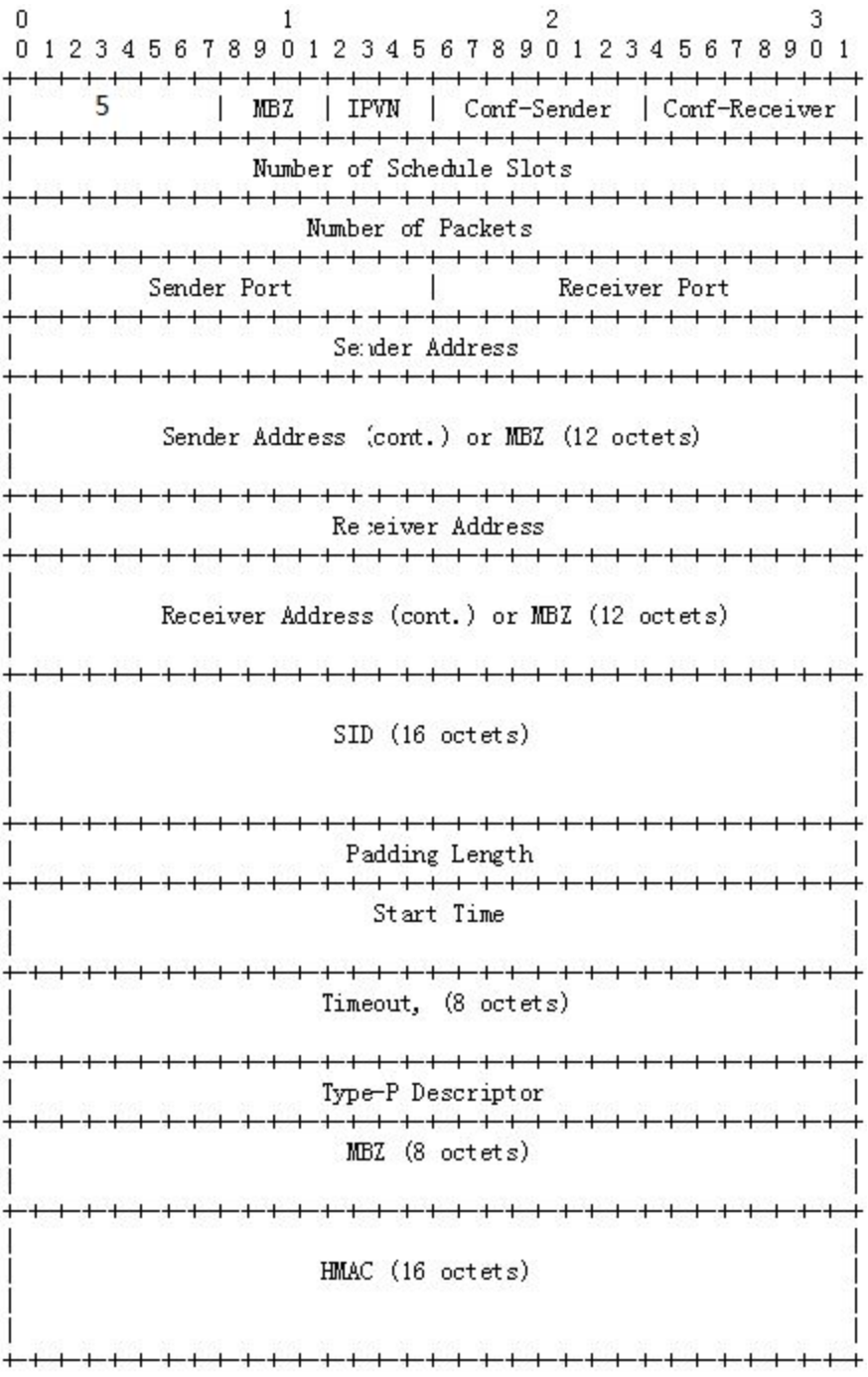

Fig 2.3.2.1 TWAMP Request-TW-Session

The first field of every command message is the Command Number whose value specifies which command the message belongs to. In this case, command number 5 indicates a Request-TW-Session command.

The IPVN indicates the type of address to be used in the address fields of this message. The values 4 and 6 indicates IPv4 address and IPv6 address respectively.

Both Conf-Sender and Conf-receiver should be set to zero as it is not up to the server to decide the UDP ports to be used for testing in TWAMP; therefore, any request received with none zero values of these fields must be responded with accept value 3.

Number of Schedule Slots and Number of Packets are not needed by the Session-Reflector as it does not produce performance metrics in TWAMP; therefore, they must be set to zero.

Sender Port and Receiver Port specify which UDP ports on the sender and reflector will be used for the testing.

Sender Address and Receiver Address indicates the address of the Session-Sender and Session-Reflector. If the addresses require more spaces to represent, the cont parts following them can be used as extension; otherwise, these fields must be set to zero. The zero value in Sender Address and Receiver Address means that the test session will use the address from the control session for the sender and reflector respectively.

The SID means session identifier which, normally, is a series of one of the IPv4 addresses from the server, a timestamp and a 4-octet random value. If no IPv4 address is available, the last 32 bits of an IPv6 address of the server will be used. As the session identifier of a session is determined by the server in TWAMP, this field within the request must be set to zero.

The Padding Length indicates how many octets will be appended to the end of the TWAMP-Test packets within this test session.

The Start Time is a timestamp format value that indicates the desired timepoint to start this test session.

The Timeout is a timestamp format value that indicates how many seconds after the receipt of a Stop-Sessions command on the reflector the test packets arrived will be possessed by it for. After that, the Session-Reflector must not reflect any test packets arrived for this session. It is worth noting that timeout is different from the SERVWAIT and REFWAIT values.

TWAMP introduced the concept of SERVWAIT and REFWAIT, which are timeout values to prevent service from stalling in control session and test session respectively. When implemented, any control session will be closed if the server does not receive any packet from it for SERVWAIT seconds, while any test session will be closed if the reflector does not receive any packet from it for REFWAIT seconds. They are complement of each other as the SERVWAIT will be paused between the receipt of the Start-Sessions and Stop-sessions commands, during which the REFWAIT is being used, but the SERVWAIT timeout will resume if the REFWAIT timeouts of all its test sessions expire; Therefore, any implementation of SERVWAIT timeout should also include the REFWAIT timeout. Normally, the default value of the SERVWAIT and REFWAIT should be 900 seconds and may be configurable.

The Type-P Descriptor indicates the QOS value to be used in the test packets. The type of value is determined by the first two bits and, normally, it will be set to zero, which means the next six bits .will be the DSCP value to be used.

After receipt of the Request-TW-Session command, the server must respond with an Accept-Session message as follows:

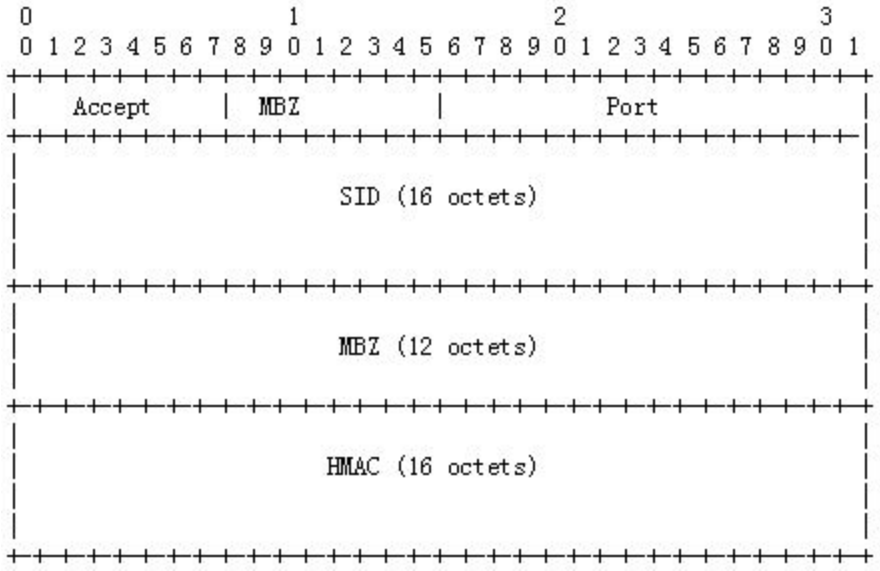

Fig 2.3.2.2 TWAMP Accept-Session

There are three different situations for the Port field. If the Accept value is 0 and the Port value is the same with the Receiver Port in the request, it means that the request has been accepted; If the Accept value is 0 and the Port value is different from the Receiver Port, then this value is a suggested port number to the reflector by the server; otherwise, the server has rejected the request and this field should be set to zero.

The SID is the unique session identifier generated by the server for this test session.

Upon receipt of the Accept-Session, the process for this test session request is deemed completed. The client may send additional requests to set up more test sessions as needed.

2.3.3 Test session start and stop

After all the test session requests have been confirmed, the client may send a Start-Sessions command with the Command Number 2 to start the test sessions, The Start-Sessions command is as follows:

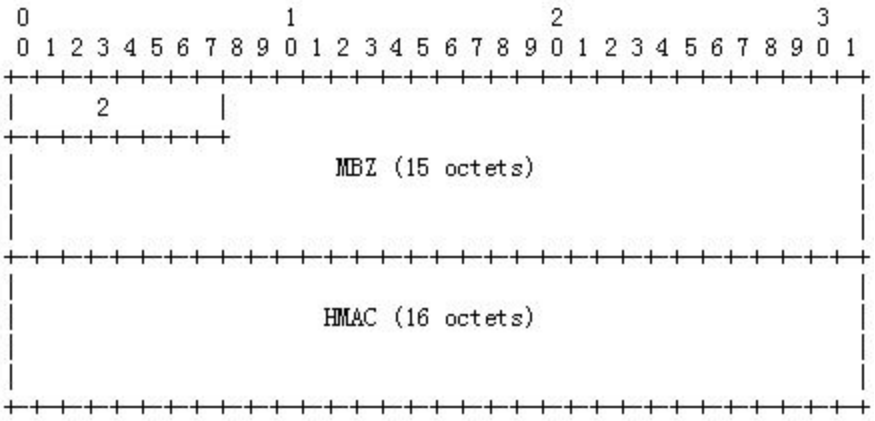

Fig 2.3.3.1 TWAMP Start-Sessions

The server must respond to this message with an Start-Ack message which has the following format:

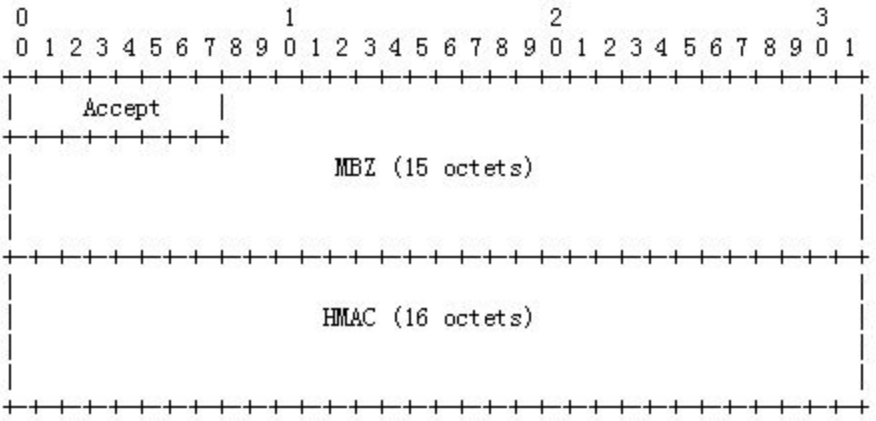

Fig 2.3.3.2 TWAMP Start-Ack

If the Accept value is 0, then this start request has been accepted and all the test sessions belonging to this control session will be started at the scheduled timepoint; otherwise, this request was rejected ,and the client should close the connection.

After the Start-Sessions command is confirmed, the only command available to the client is the Stop-Sessions command.

When it is appropriate, the client should send the Stop-Sessions command with the Command Number 3 to end all the test sessions. The Stop-Sessions command is as follows:

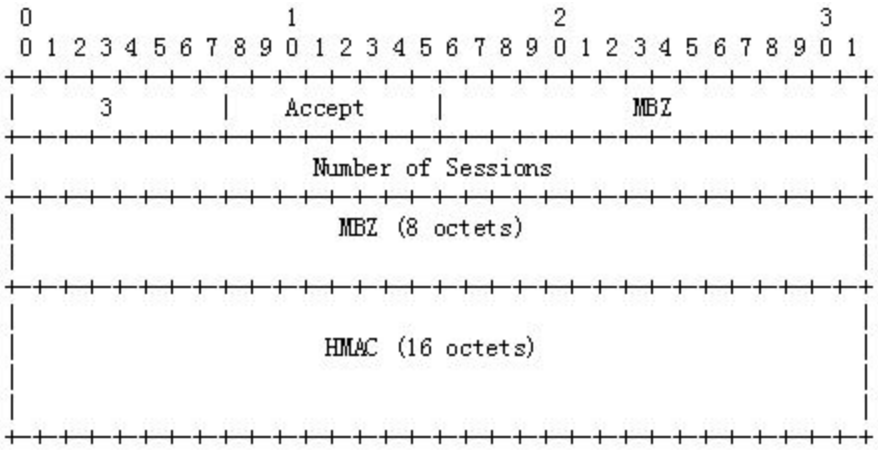

Fig 2.3.3.3 TWAMP Stop-Sessions

If the Accept value is not zero, it indicates an abnormal termination of the test; all the metrics acquired from this control session should be deemed invalid.

The Number of Sessions is the number of test sessions belonging to this control session. It is used as an authentication tool to ensure data integrity and security. If the value does not match with the number of currently active test sessions, then the server should close this control session and all its test sessions, and any metrics acquired from this control session should be deemed invalid.

After the stop request has been sent, this metric test is deemed completed, and the client may set up new test sessions and start another test as needed.

2.4 TWAMP-Test

As the Session-Sender and the Session-Reflector are working independently during the test period, this report will introduce them separately.

2.4.1 Session-Sender

When it is appropriate, the sender will send out the UDP test packets with TTL/HopLimit 255 to the reflector. The format of the test packet is as follows:

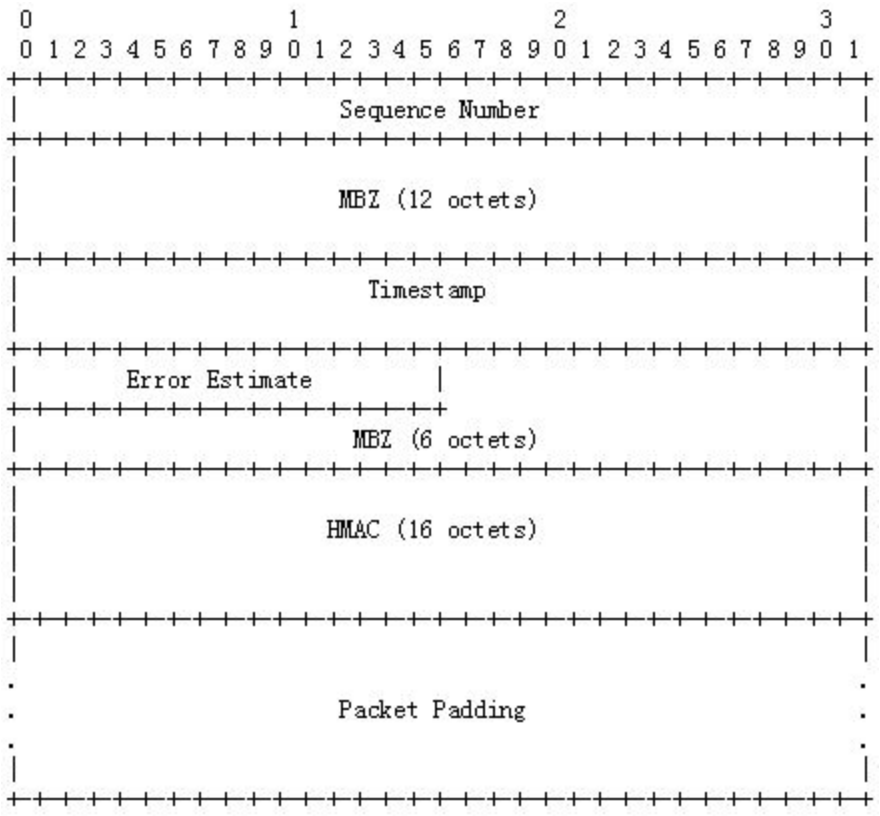

Fig 2.4.1.1 TWAMP test packet

The Sequence Number is a session-unique sequence value startinging from zero and incremented by one for each packet.

The Timestamp is the best approximation of the departure time of the packet.

The Error Estimate is a common field that indicates the estimate of timing errors, and it is as follows:

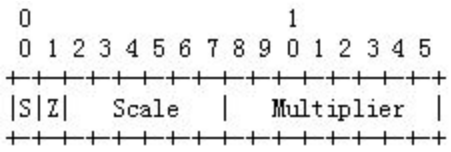

Fig 2.4.1.2 TWAMP Error Estimate

The S field will be set if the sender of this packet is synchronized to UTC with an external source.

The Z field is a MBZ field.

Scale and Multiplier represent the value of the estimate; the estimate is equal to Multiplier*2^(-32)*2^Scale (seconds). It should be noted that the Multiplier should be greater than zero; otherwise, this packet should be deemed invalid and be discarded.

Normally, the Packet Padding should be randomly generated; however, it is possible to set it as all zero.

Besides of sending the test packets, the sender will also be ready to receive the responses from the reflector and collect data from them for metric computation.

2.4.2 Session-Reflector

After the test sessions have started, the reflector will be ready to reflect test packets back to the sender. When a test packet arrives, the reflector will first record the arrival time of the packet, then generate a reflected packet with the information from it and send it out as an UDP packet with TTL/HopLimit set to 255.

The reflected packet contains the information needed for the metric computation by the sender and has the following format:

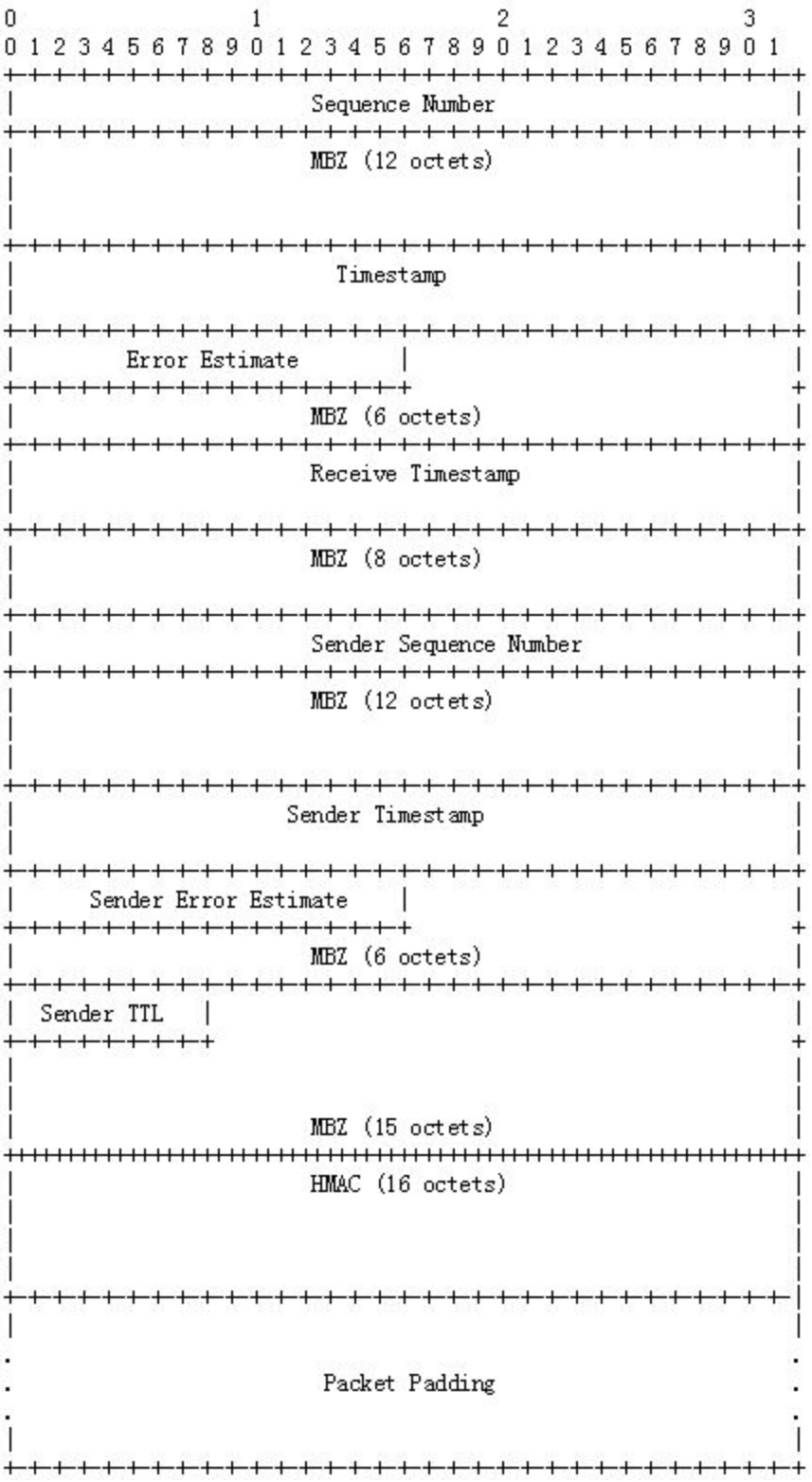

Fig 2.4.2.1 TWAMP reflected packet

The Sequence Number is a session-unique sequence value starting from zero and incremented by one for each reflected packet. It is a value for the reflected packet which is independent from the sequence value in the received packet.

The Timestamp is the best approximation of the departure time of the reflected packet.

The Error Estimate is the error estimate for the reflected packet, which mostly applies to the Timestamp and Receive Timestamp field of the packet..

The Receive Timestamp indicates the time when the corresponding test packet arrived.

The Sender Sequence Number, Sender Timestamp and Sender Error Estimate are values copied from the corresponding fields of the received test packet.

The Sender TTL is a value copied from the TTL/HopLimit field in the IP header of the received packet. Normally, If the value in the header can not be read, this field will be set to 255.

Normally, the Packet Padding should be randomly generated; however, it is possible to set it as all zero. The reflector may reuse the padding from the received packet, and if the size is too large, the reflector should drop the bits in the highests number positions.

It is worth noting that the sender and reflector can use the padding on both packets they send to achieve equal IP payload lengths of those packets.

After the stop command is received, the reflector will ignore the follow-up test packets for the corresponding test sessions after the Timeout seconds.

2.5 Integrity and Security

There are three modes available in TWAMP: Mode value 1 for unauthenticated, Mode value 2 for authenticated and Mode value 4 for encrypted.

Regarding the data security during the process, TWAMP has introduced several measures in authenticated and encrypted modes towards three different phases.

2.5.1 Connection Setup

Both the authenticated and encrypted modes use the same measures in this phase.

During the initial exchange between the client and the server, the KeyID, Salt, and Count are used to protect the Token field. In this case, Token is encrypted by AES CBC mode with the secret key as the key and zero as the IV.

To obtain the secret key, the client and the server will use the same mapping with the KeyID to get the same shared secret, which is a passphrase encoded with the ASCII NVT [RFC5198]. The shared secret must not contain newlines and should avoid control characters.

After that, they will generate the secret key from the shared secret using the PBKDF2 function [RFC2898] with the Salt and Count from the Server-Greeting message. The PRF for this function is HMAC-SHA1 [RFC2104].

The server will decrypt the Token field and check for the challenge after the receipt of the Set-Up-Response. If the check goes through, then this packet from the client is deemed valid, and the server will perform further actions with it,

By using the encryption for the Token, TWAMP protects the session-keys within it as well as authenticates the identity of the client in the very first place.

2.5.2 TWAMP-Control

Both the authenticated and encrypted modes also utilize the same measures under this phase.

For the control session, TWAMP utilizes AES CBC mode to protect its communications from security attacks. The encryption of packets is a stream encryption, which uses the TWAMP-Control AES Session-key from the Set-Up-Resoponse as the key with the IV of the receiver of this packet.

If the server accepts the control connection, then the encryption will start with the Start-Time field of the Server-Start message, and every packet within this control session afterwards will be encrypted.

2.5.3 TWAMP-Test

In TWAMP, the mode of the control session is inherited to its test sessions.

For the test sessions, the authenticated mode and encrypted perform different actions to protect the packets.

To protect the data security of packets belonging to a test session, the encrypted mode will encrypt all the sequence number, the timestamp, and the error estimate portions of the packet to achieve maximum data security and conficiency.

In the other hand, the authenticated mode will only encrypt the sequence number portion of the packet, leaving the timestamp and error estimate portions to be plaint-text, in order to gain faster processing speed and more accurate timestamp for departure, since the time needed for the encryption is noticeable for the performance measurement.

In authenticated mode, the sequence number portion (the first 16 octets) is encrypted with AES ECB mode, whereas the sequence number, timestamp, and error estimate portions (the first 32 octets for test packet and 96 octets for reflected packet) are encrypted by the AES CBC mode with a zero IV under encrypted mode.

The key to be used for the encryption is obtained by encrypting the Session-key for the control session. AES single-block ECB mode is used with the SID of the test session as the key.

It should be pointed out that although the test sessions are conducted under the UDP protocol, the lost, duplication, or reordering of test packets will not cause problems to decrypt other packets for the reason that no chaining between packets is created with the encryption under authenticated and encrypted modes.

2.5.4 Integrity Protection

In authenticated and encrypted modes, TWAMP uses HMAC-SHA1 truncated to 128 bits to protect the data integrity during both control and test sessions, and the authentication of the message (i.e. the calculation of HMAC) always happens before the encryption.

For the control sessions, the HMAC field will be appended to the end of every packets within the session after the connection setup phase and covers every other fields of the packet; however, one special case could be the HMAC in the first Accept-Session message which also covers the Start-Time field of the previous Server-Start message. The HMAC is computed with the HMAC Session-key from the Set-Up-Response message.

For the test sessions, the HMAC field is appended to the end of every packets within the sessions and covers the same portions of the packet as the encryption. The key is acquired by encrypting the TWAMP-Control HMAC Session-key with AES two-block CBC mode, using the SID of the test session as the key with a zero IV.

During the test sessions under authenticated and encrypted modes, the reflector will decrypt the received packet after the arrival time has been recorded, and then check the integrity of it before starting to extract information from the packet.

It should be pointed out that the HMAC is encrypted with the packet in the control sessions while sent in clear text in the test sessions.

If unauthenticated mode is being used, then neither the authentication nor the encryption will be performed. In this case, the relating fields will be unused and the HMAC field should be set to zero; moreover, the test and reflected packets within test sessions will be simplified to the following:

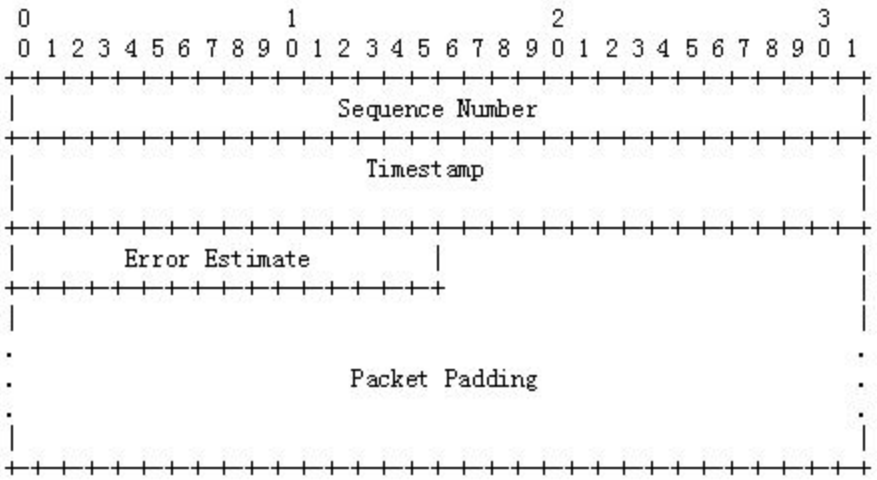

Fig 2.5.4.1 the simplified test packet

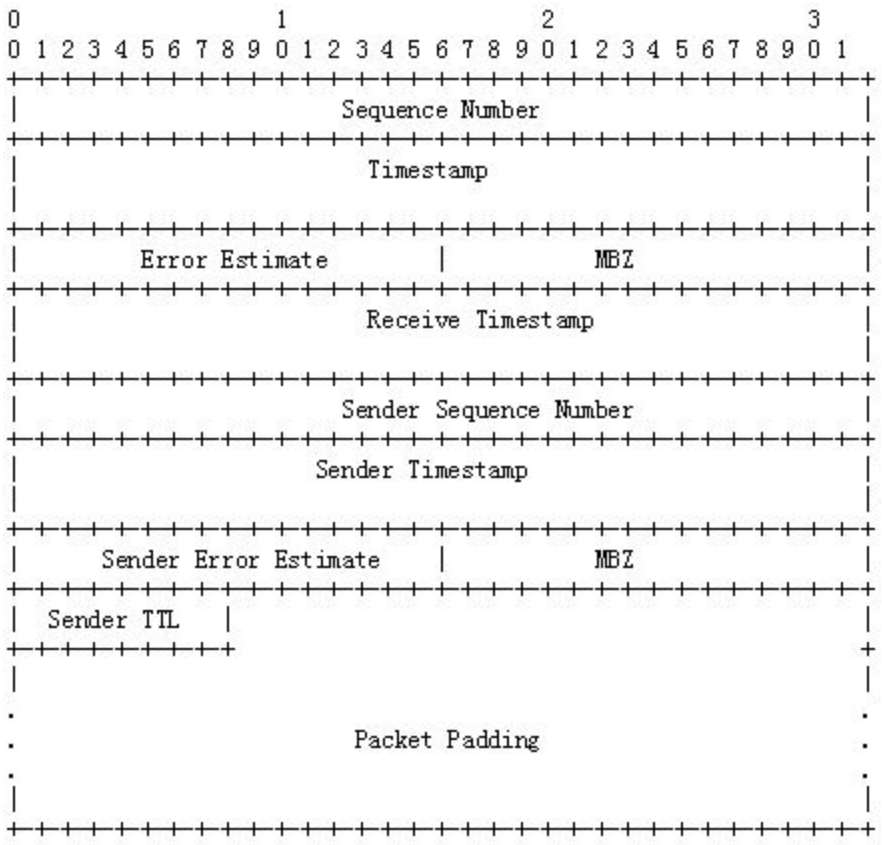

Fig 2.5.4.2 the simplified reflected packet

3. Y.1731

OAM Functions And Mechanisms For Ethernet-based Networks (Y.1731) is an OAM standard maintained by ITU-T. It covers both fault management and performance monitoring areas for ethernet networks.

3.1 Logical Model

There are several roles within this model: ME, MEG, MEP, and MIP.

ME: maintenance entity indicates the connectivity between two flow points of a connection and is the basic element for OAM in Y.1731. MEs can be nested but not overlapped.

MEG: ME group is a group of MEs belonging to the same connection of this MEG and within the same administrative boundary of this MEG.

MEG level is defined in the networks with nested MEGs for easy management purpose. In this case, each MEG possesses a MEG level, and all MEs of it will inherit this level. Higher level MEs have the transparency over lower MEs.

There are eight MEG levels available, from 0 to 7, and they can be used independently within different administrative boundaries or shared amongst them when necessary.

In the situation with customers, providers and operators, the default shared MEG level assignment would be: 0, 1 and 2 for the operator; 3 and 4 for the provider; 5, 6 and 7 for the customer.

MEP: MEG end point is the end points of the MEs within this MEG,

MIP: MEG intermediate point is a flow point which only processes some kinds of OAM frames; otherwise, it will just relay every frame received.

Below is an illustration of basic MEs in a simple P2P connection.

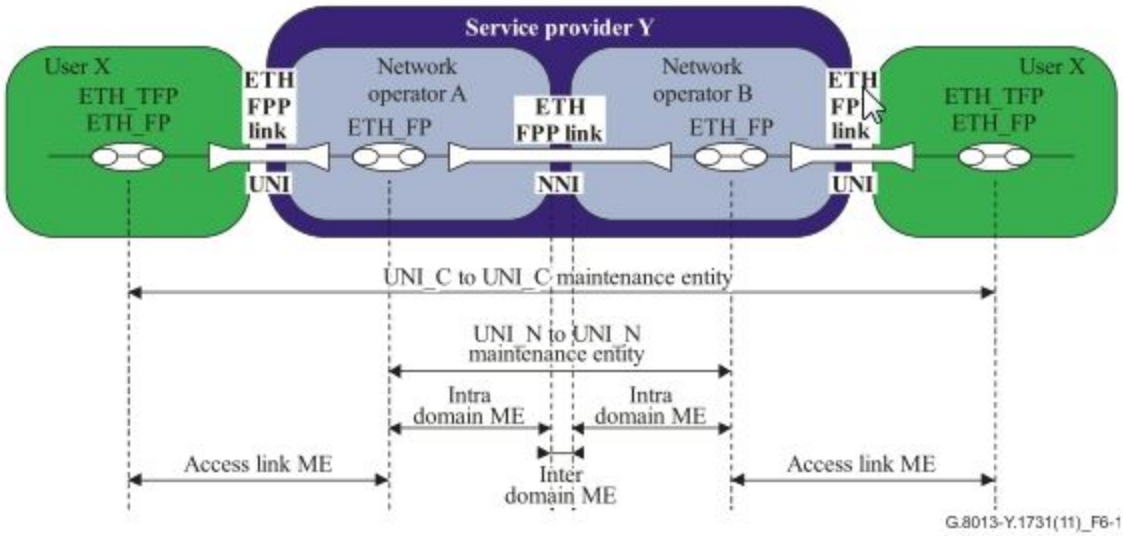

Fig 3.1.1 simple P2P connection

In the example, we have four administrative domains: User X, Provider Y, Operator A and Operator B.

Within User X domain, there is the UNI_C to UNI_C ME.

Within Provider Y domain,there is the UNI_N to UNI_N ME.

Each Operator A and B domains has an intra domain ME.

Amongst all these domains, there are several inter domain MEs (i.e. two access link MEs and one inter domain ME between two Operator domains).

Based on the logical model stated above, Several OAM functions are defined to maintain the network.

3.2 Fault Management

Fault management in Y.1731 is to detect, verify, localize and notify the defect conditions within the ethernet network. This part of the standard is realized with functions as follows.

3.2.1 ETH-CC

Ethernet continuity check function is used to detect the continuity status of MEs within the network through MEPs sending periodic hello packets (CCMs) to other MEPs of the same MEG.

CCM is the PDU of this function and contains ETH-CC information; the frame contains this PDU is a CCM frame and is drop ineligible.

In order to start this function, each MEP of this MEG will require the following information:

MEG ID: the identifier of the MEG this MEP belongs to.
MEP ID: the identifier of this MEP within the MEG.
List of peer MEP IDs: a list includes the MEP IDs of every legal peer MEP within this MEG.
MEG level: the MEG level of this MEP.
Priority: the priority of the CCM frames sent. CCM frames have the highest priority of the data flows.
ETH-CC transmission period: the length of time between each CCM frame sent out by the MEPs.

The period of the CCM frames within a MEG should be the same and could be one of the following: 3.33ms (the default period for protection switching application), 10ms, 100ms (the default period for performance monitoring application), 1s (the default period for fault management application), 10s, 1min and 10min.

After the receipt of a CCM frame, the following defect conditions can be detected:

LOC: loss of continuity within this ME is detected if no CCM frame associated is received for 3.5 times the period of this MEG.
Unexpected MEG level: unexpected MEG level is detected if the MEG level within the frame is lower than the receiver's MEG level.

Mismerge: a mismerge is detected if the frame has the right MEG level but not a correct MEG ID.

Unexpected MEP: unexpected MEP is detected if the frame has the right MEG level and MEG ID but not a legal MEP ID within the list in the receiver.

Unexpected period: unexpected period is detected if the frame has the right MEG level, MEG ID and MEP ID, but not the same period with the one of the receiver.

3.2.2 ETH-LB

Ethernet loopback function is used for connectivity and diagnostics tests when needed, and provides two types of testing through a simple request/reply mechanism: Unicast or Multicast.

The PDU for this function is LBM for the request and LBR for the reply.

For unicast ETH-LB, one of the following two modes can be deployed when needed: a connectivity check between a MEP and a MIP or a peer MEP; or a diagnostics test between two MEPs.

It should be pointed out that these are not compatible and only one mode can be running at a time.

Under unicast ETH-LB, the associated MEPs will request the following information:

MEG level: the MEG level of this MEP.
Destination unicast MAC address: the address of the destination.
Data: data is an optional element to provide availability for test data usage when needed.
Priority: the priority of the ETH-LB frames sent.
Drop eligibility: the eligibility to drop the frame when congestion occurs.

Moreover, the associated MIP will need to know the MEG level of them.

When needed, the initiating MEP under this mode will send out a unicast LBM frame with a unique Transaction ID/ Sequence Number to a MIP or a peer MEP as a request for ETH-LB reply. The transaction ID should be unique and must not repeat with the same MEP within one minute.

When a MEP receives a unicast LBM frame in this mode, it will first check the frame. If the frame has the correct MEG level, then it is valid, and the MEP will send an LBR frame, which is the received LBM frame with corrected source and destination addresses and OpCode, back to the initiating MEP.

The connectivity between the two test objects will be deemed lost, If the initiating MEP does not receive the corresponding LBR frame within 5s after the unicast LBM frame was sent.

The connectivity test can be performed with different frame sizes. In this case, the Data TLV field can be used to provide different payload lengths.

In diagnostics test mode, the initiating MEP will send out unicast LBM frames with test patterns for verification to a peer MEP.

The test patterns is generated by a test signal generator associated with the MEP and contained within the Test TLV field of the LBM; it will also be verified by the test signal receiver at the receiver side.

When a MEP receives a unicast LBM frame in this mode, it will check the MEG level within the frame and verify the test TLV with the test signal receiver.

In the case of out-of-service diagnostics test deployment, the MEPs will send out lock signals within the immediate client MEG level to inform the change, since no data traffic will be allowed on the diagnosed ME.

In both modes, any LBR frame received by a MIP should be deemed invalid and discarded.

Under multicast ETH-LB, the connectivity between a MEP and peer MEPs will be tested; and the ETH-LB frames are drop ineligible.

In this mode, the associated MEPs will require the following information:

MEG level: the MEG level of this MEP.
Priority: the priority of ETH-LB frames sent.

When needed, the initiating MEP will send a multicast LBM frame towards all its peer MEPs with a transaction ID.

When a multicast LBM frame is received by a MEP, it will first be checked for the MEG level. If the frame is valid, then the MEP will send an LBR frame, which is the received LBM frame with corrected source and destination addresses and OpCode, back to the initiating MEP, with a random delay between 0 and 1 second.

If the corresponding LBR frame is not received by the initiating MEP within 5s after sending the multicast LBM frame, then the connectivity between these two MEPs is deemed lost.

After the test, the initiating MEP will return a list of peer MEPs with connectivity to it.

It should be noted that any LBR frame received without a correct transaction ID or MEG level, or without the time limit, is deemed invalid and should be discarded; and any LBR frame received by a MIP is invalid and should be discarded.

3.2.3 ETH-LT

Ethernet link trace function is used to trace the link towards a given MAC address when needed through a series of route trace request and replies.

The PDU for this function is LTM for the request, and LTR for the reply; the corresponding frames are drop ineligible.

When using this function, the associated MEP will require the following information:

MEG level: the MEG level of this MEP.
Priority: the priority of the ETH-LT frames sent.
Target MAC address: the MAC address of the destination of the request.
TTL: time to live value of the LTM frames sent. A frame with a TTL equals to 1 will not be forwarded.

Moreover, the associated MIPs will need to know the MEG level of them.

When needed, the initiating MEP will send an LTM frame with a unique transaction ID towards the target MAC address. The frame also has an LTM egress identifier TLV indicating the source of it.

When a MEP or MIP receives a LTM frame, it will be forwarded to the network element's ETH-LT responder to check for the MEG level, TTL and the LTM egress identifier TLV. If the frame is valid and the target MAC address is within this network element or within the MAC table with an egress port different from the ingress port of this frame, the MIP or MEP will send an LTR frame back to the initiating MEP with a randomized delay between 0 and 1s.

The LTR frame has a LTR egress identifier TLV which has a Last Egress Identifier field and a Next Egress Identifier field. The Last Egress Identifier field contains the LTM egress identifier TLV of the received frame while the Next Egress Identifier field indicates the sender of this frame. The frame also contains a TLV to describe the receiving port of the LTM frame; the TLV will be the reply ingress TLV or the reply egress TLV depending on whether the receiving port is an ingress port or egress.

Moreover, if the LTM frame is received by a MIP with an egress port for the target MAC address different from the ingress port of the reception, it will forward the LTM frame with the

corrected TTL, source address and LTM egress identifier TLV. In the case, the FwdYes bit of the Flags field within the LTR frame will be set.

If a MEP receives an LTM frame not targeted to it, the MEP will ignore the frame and discard it.

A good practice for ETH-LT is to initiate a ETH-LB from the initiating MEP towards the target right before sending the LTM frame to let the associated MIPs and MEP be aware of the target.

When the initiating MEP receives an LTR frame, the frame will be valid only if it has the correct transaction ID and arrives within 5s after sending the corresponding LTM frame; otherwise, it should be ignored and discarded.

Any LTR frame received by a MIP is invalid and should be discarded.

A list of MAC addresses of intermediate points along the link towards the target will be returned when the link trace is completed.

3.2.4 ETH-AIS

Ethernet alarm indication signal function is used to suppress alarms in client layer for server layer detected defect conditions through periodic indication signals.

The PDU used by this function is AIS; and the AIS frame is drop ineligible.

When using this function, the following information is required by the associated MEPs to send AIS frames:

Client MEG level: the MEG level of the immediate client layer.
ETH-AIS transmission period: the length of time between AIS frames sent.
Priority: the priority of the AIS frame.

Moreover, the associated MEPs will need the local MEG level in order to receive a AIS frame.

When enabled, a MEP will send out AIS frames to other domains if a defect condition is detected. The AIS frames will be sent at periodic way as long as the defects still exist. The default value for the period is 1s; and the value of the period will be conveyed along the frame. The MEP must send the AIS frames to every client MEPs within 1s.

When a MEP receives an AIS frame, it will first check the MEG level of the frame. If it goes through, then the defect condition from the frame is accepted and the LOC alarms of corresponding peer MEPs will be suppressed. If no AIS frame is received after 3.5 times the period, then this suppression is cleared; a MEP will also clear the suppression if a new defect is detected outside the defect condition from the frame.

On a multipoint connection, the MEP received the AIS frame will suppress alarms for every peer MEPs since it can not identify the MEPs with defect conditions.

3.2.5 ETH-RDI

Ethernet remote defect indication function is used by a MEP to inform peer MEPs when ETH-CC detects defect conditions.

This function utilize the PDU of ETH-CC to convey the defect conditions to peer MEPs.

In order to utilize this function, the following information will be needed by the associated MEPs:

MEG level: the MEG level of this MEP.
ETH-RDI transmission period: the period of the CCM frame.
Priority: the priority of the CCM frame.

When ETH-CC of a MEP detects a continuity lost, the MEP will put the fault information within the RDI field of every CCM frame it sends afterwards to inform peer MEPs until the lost has been solved.

Once a CCM frame arrives to a MEP and passes the its frame check, the MEP will check the RDI field and acquire the defect information in it, if any. Once the defect condition is acquired, it will not be cleared until the CCM frames received from all peer MEPs clear that condition.

3.2.6 ETH-LCK

Ethernet locked signal function is used to inform client layer MEPs with administrative interruption of the service for specific server MEPs through periodic transmission of lock signals. It has similar effect to suppress the alarms as ETH-AIS, whereas it is focus on administrative interruptions and ETH-AIS is focus on detected defect conditions.

The PDU of this function is LCK. The frame contains this PDU is LCK frame and is drop ineligible.

In order to send LCK frames, a MEP needs the following information:

Client MEG level: MEG level of the immediate client layer.
ETH-LCK transmission period: the transmission period of LCK frames.
Priority: the priority of the LCK frames.

Moreover, the associated MEPs will need a local MEG level to receive LCK frames.

When a MEP is locked by the administrator, it will send periodic LCK frames towards the client MEPs. The period of ETH-LCK should be the same with the period of ETH-AIS; and the first LCK frame is sent immediately after getting the locked state. Therefore, just like ETH-AIS, the first LCK frames should be sent to every client MEP within the first second after the defect occurs.

A MEP receiving an LCK frame will first check the MEG level of the frame. If the frame passes the check, then the lock condition of the corresponding MEP will be set. This condition will stay active until 3.5 times the period without any LCK frames from that MEP received.

3.2.7 ETH-Test

Ethernet test signal function is used to perform one-way one-demand diagnostics test with ETH-LB. Various metrics can be acquired from this function.

The PDU of this function is TST.

In order to utilize this function, the associated MEPs need the following information:

MEG level: the MEG level of this MEP.
Unicast destination MAC address: the address of the target peer MEP.
Data: optional element for some of the tests.
Priority: the priority of TST frames.

For out-of-service test, the initiating MEP will send TST frames to the target peer MEP; and both of them will send periodic LCK frames to client MEPs.

For in-service test, the initiating MEP will only use a predefined section of the bandwidth for sending the TST frames.

3.2.8 ETH-APS

Ethernet automatic protection switching function is used to control protection switching operations for enhanced reliability.

The PDU of this function is APS.

3.2.9 ETH-MCC

Ethernet maintenance communication channel function is used to convey management instructions between MEPs.

The PDU of this function is MCC.

In order to utilize the function, the associated MEPs need the following information:

MEG level: the MEG level of the MEP
Unicast destination MAC address: the address of the target.
OUI: organizationally unique identifier is an identifier indicating the organization of an ETH-MCC implementation.
Data: optional element.
Priority: the priority of the MCC frame.

3.2.10 ETH-EXP

Ethernet experimental OAM function is used for temporary OAM function experiment.

The PDU of this function is EXM and EXR.

Normally, this function should be used within one single administrative domain for testing new OAM functionality.

3.2.11 ETH-VSP

Ethernet vendor-specific OAM function is used to implement vendor-specific OAM functions.

The PDU of this function is VSM and VSR.

3.2.12 ETH-CSF

Ethernet client signal fail function is used by a MEP on a P2P connection to inform the peer MEP with the client defect condition when no other proper measures are supported.

The PDU of this function is CSF.

Under this mode, when a MEP detects a defect condition indication from an ingress client port, it will send periodic CSF frames to the peer MEP. This frame can indicate one of the following three types of defect:

C-LOS: client loss of signal.
C-FDI: client forward defect indication.
C-RDI: client reverse defect indication.

When a valid CSF frame with defect condition indication is received by the peer MEP, the client defect condition is set until one of the following situations: no CSF frame arrives within a set period of time; or a CSF frame is received with a defect clear indication (C-DCI).

3.2.13 ETH-BN

Ethernet bandwidth notification function is used to notify client MEPs of the bandwidth status within the server network through periodic notifications.

The PDU of this function is BNM. The BNM frame is periodic and drop ineligible.

In order to utilize this function, the associated server MEPs need the following information:

Client MEG level: the MEG level of the immediate client layer.
ETH-BN transmission period: the period of the BNM frames.
Hold time: the buffer period threshold to filter short changes.
Priority: the priority of the BNM frames.
Port ID: a 32 bit unique port identifier which will be useful under multipoint MEG situation.

Moreover, the associated client MEPs will need to have a local MEG level configured.

When a server MEP detects a stable bandwidth change, it will send periodic BNM frames to the client MEPs on its side.

When a client MEP receives a valid BNM frame, it will forward the source MAC, Port ID and bandwidth information to its management system for further procession.

3.2.14 ETH-ED

Ethernet expected defect function is used by a MEP to inform peer MEPs when an interruption of its CCM transmission occurs, but no data traffic will be affected. In this case, the CCM detection for that MEP on the peer MEPs will be temporarily suspended.

The PDU of this function is EDM. The EDM frame is drop ineligible.

In order to utilize this function, the MEP with interruption should have the following information:

MEG level: the MEG level of the MEP.
MEP ID: the identifier of the MEP.
Expected defect duration: the period of detection suspension.
ETH-ED transmission period: the period of the EDM frame.
Priority: the priority of the EDM frame.

Moreover, the peer MEPs will need to have a local MEG level to receive EDM frames.

When a MEP expects an upcoming interruption for its CCM transmission, which will not affect the data traffic, it will send out periodic EDM frames with its MEP ID and expected duration to peer MEPs.

The MEPs which receive the valid EDM frames will pass the information of the frame to the element management function (EMF) for the detection suspension. The suspension will last for the time of the Expected Defect Duration from the frame.

3.3 Performance Monitoring

Performance monitoring in Y.1731 is to measure performance metrics with various functions in ethernet network.

3.3.1 Performance Metrics

The following performance metrics will be covered under this protocol. It should be noted that every frame used under a certain performance monitoring should be in the same CoS, possess the same priority, and be drop ineligible.

Frame loss ratio: frame loss ratio is the percentage of lost frames over the frames sent within a certain period. For a MEP on a ME, the near-end frame loss refers to frames sent towards it, while the far-end frame loss refers to frames sent from it.

Frame delay: the one-way frame delay is the time past from the start of sending the first bit of the frame by the source host to the end of receiving the last bit of it by the destination host. The two-way frame delay is the time past from the start of sending the first bit of the frame by the source host, which will be responded by the destination host, to the end of receiving the last bit of the response by the same host. It should be noted that synchronization of clocks between associated hosts is necessary for one-way frame delay measurement.

Frame delay variation: the frame delay variation is the difference between the results of two frame delay measurements of one specific frame flow. Clock synchronization is not needed for delay variation measurement.

3.3.2 ETH-LM

Ethernet loss measurement function is used to gather frame loss information on peer MEPs.

Based on the amount of MEPs conducting the measurement, two modes of this function are available: single-ended for loss measurement on one side of the ME and dual-ended for measurement on both sides.

In this function, the measurement of frame loss is done through monitoring the transmission and reception data on the associated MEPs. As a result, counters for sending and receiving of frames towards different MEPs and with different priorities will be implemented on those MEPs. The two basic counters for this function are TxFC1, which counts for the number of data frames sent, and RxFC1, which counts for the number of data frames received.

The rules regarding to counting the OAM frames are as follows:

OAM frames for proactive functions used by termination functions are counted only under single-ended mode.

In both modes, OAM frames for adaptation functions are counted while frames for on-demand functions are not counted.

In single-ended mode, the PDU used is LMM for information request and LMR for information reply. Both the LMM frame and LMR frame are drop ineligible.

LMM frames contain a **TxFCf** element which is the TxFC1 value at the transmission time.

LMR frames contain the following elements:

TxFCf: the TxFCf field of the corresponding LMM frame.
RxFCf: the RxFC1 value at the corresponding LMM frame reception time.
TxFCb: the TxFC1 value at the LMR transmission time.

In this mode, a initiating MEP sends periodic LMM frames to the peer MEP to request the counter values on it and waits for response from the it.

When a MEP receives a valid LMM frame, it will reply the initiating MEP with an LMR frame.

After receipt of the LMR response, the frame loss can be calculated with the received information as follows:

Frame Loss$_{\textbf{far-end}}$ = | TxFCf[t_c] - TxFCf[t_p] | - | RxFCf[t_c] - RxFCf[t_p] |
Frame Loss$_{\textbf{near-end}}$ = | TxFCb[t_c] - TxFCb[t_p] | - | RxFC1[t_c] - RxFC1[t_p] |
t_c means at the time of this LMR frame receipt, and t_p means at the time of the last LMR frame receipt.

In dual-ended mode, the PDU used is CCM with the following elements:

TxFCf: the TxFC1 value at the CCM transmission time.
RxFCb: the RxFC1 value at the last CCM frame reception time.
TxFCb: the TxFCf field of the last CCM frame from the peer MEP.

Under this mode, the associated MEPs send periodic ETH-CC CCM frames with the above ETH-LM information to peer MEPs.

After receipt of the CCM frame, the frame loss can be calculated with the received information as follows:

Frame Loss$_{\textbf{far-end}}$ = | TxFCb[t_c] - TxFCb[t_p] | - | RxFCb[t_c] - RxFCb[t_p] |
Frame Loss$_{\textbf{near-end}}$ = | TxFCf[t_c] - TxFCf[t_p] | - | RxFC1[t_c] - RxFC1[t_p] |
t_c means at the time of this CCM frame receipt, and t_p means at the time of the last CCM frame receipt.

It should be pointed out that if LOC is detected with a peer MEP, then the result of frame loss measurement of that MEP will be replaced with 100% loss ratio.

3.3.3 ETH-DM

Ethernet frame delay measurement is used to produce frame delay information of the tested ME.

In order to utilize this function, the following information is needed by the associated MEPs:

MEG level: the MEG level of this MEP.
Unicast destination MAC address: the address of the peer MEP on the tested ME.
DM application: it indicates the application of this test.(proactive or on-demand)
Data: optional element when needed.
Priority: the priority of the ETH-DM frames.
Test ID: optional element for test session identification. It should be unique amongst measurements of the same kind within the MEG.

The ETH-DM frames are drop ineligible.

Based on the amount of MEPs conducting the measurement, two modes of this function are available: single-ended for delay measurement on one side of the ME and dual-ended for measurement on both sides.

In single-ended mode, the PDU used is DMM for the request and DMR for the response.

DMM frames contain a **TxTimeStampf** field which is the timestamp of the DMM transmission.

DMR frames contain the following elements:

TxTimeStampf: the same field from the DMM received.
RxTimeStampf: the optional timestamp of the DMM frame reception.
TxTimeStampb: the optional timestamp of the DMR transmission.

Under this mode, a initiating MEP sends periodic DMM frames to peer MEPs to request the two-way delay information and waits for the response.

When a MEP receives a valid DMM frame, it will send a DMR reply back to the initiating MEP.

After the receipt of the DMR frame, the two-way frame delay can be calculated with the received information as follows:

Frame Delay$_{\text{two-way}}$ = RxTimeb - TxTimeStampf
RxTimeb is the time of DMR reception.

If TxTimeStampb and RxTimeStampf options are included in the DMR and the clocks are synchronized, then one-way frame delay can also be calculated, and the result of two-way frame delay will be more accurate. With them included, the frame delay is as follows:

Frame Delay$_{\text{two-way}}$ = (RxTimeb - TxTimeStampf) - (TxTimeStampb - RxTimeStampf)
Frame Delay$_{\text{one-way\_far}}$ = RxTimeStampf - TxTimeStampf
Frame Delay$_{\text{one-way\_near}}$ = RxTimeb - TxTimeStampb

In the dual-ended mode, the PDU used is 1DM. 1DM frame contains TxTimeStampf which is the timestamp of the 1DM transmission.

Under this mode, the associated MEPs will send periodic 1DM frames with one-way delay information to peer MEPs.

After the receipt of the 1DM frame, the one-way frame delay can be calculated with the received information as follows:

Frame Delay$_{\text{one-way}}$ = RxTimef - TxTimeStampf
RxTimef is the timestamp of the 1DM reception.

3.3.4 ETH-SLM

Ethernet synthetic loss measurement function is used for frame loss measurement with the self-generated traffic (synthetic frame). It is similar towards ETH-LM function in various ways, but it utilizes its PDU as the traffic data.

Similar to ETH-LM, single-ended and dual-ended modes are available for this function.

Two counters are set for synthetic frames of specific test ID, with specific CoS and from specific peer MEP: TxFC1 for transmission and RxFC1 for reception.

In order to utilize this function, the following information is needed by the associated MEPs:

MEG level: the MEG level of the MEP.

Data: optional element.
Destination MAC address: the address of the peer MEP.
Test ID: optional session identifier which is unique within the tested MEG.
Priority: the priority of the synthetic frame.

The measurement period is defined as the period of time for which this function can get the best approximation of frame loss with scheduled synthetic frames.

In single-ended mode, the PDU used is SLM for request and SLR for reply. They both contain the following elements:

Test ID: the test ID of the session.
MEP ID: the identifier of this MEP within the MEG.
TxFCf: the value of TxFC1 on the initiating MEP at the SLM transmission time.
TxFCb: the value of RxFC1 at *the SLR transmission time*. It is set to zero in SLM.

Under this mode, the initiating MEP sends periodic SLM frames to peer MEPs within the measurement period and waits for SLR replies.

When a MEP receives a valid SLM frame, it will respond with an SLR frame.

At the end of the measurement period, the initiating MEP will calculate the frame loss as follows:

Frame loss$_{\textbf{far-end}}$ = | TxFCf[t_c] - TxFCf[t_p] | - | TxFCb[t_c] - TxFCb[t_p] |
Frame loss$_{\textbf{near-end}}$ = | TxFCb[t_c] - TxFCb[t_p] | - | RxFC1[t_c] - RxFC1[t_p] |
t_c is the end time of the period while t_p is the time of first reception of SLR for TxFCf and TxFCb, *and is the start time of the period for RxFC1*.

In dual-ended mode, the PDU used is 1SL which contains the following elements:

Test ID: the session identifier.
MEP ID: the identifier of this MEP.
TxFCf: the value of TxFC1 at the 1SL transmission time.

Under this mode, the initiating MEPs send periodic 1SL frames to peer MEPs within the measurement period.

At the end of the measurement period, the frame loss is calculated as follows:

Frame loss$_{\textbf{near-end}}$ = | TxFCf[t_c] - TxFCf[t_p] | - | RxFC1[t_c] - RxFC1[t_p] |
t_c is the end time of the period while t_p is the time of first reception of 1SL for TxFCf, *and is the start time of the period for RxFC1*.

3.5 Example of Implementation

An example of implementation is as below:

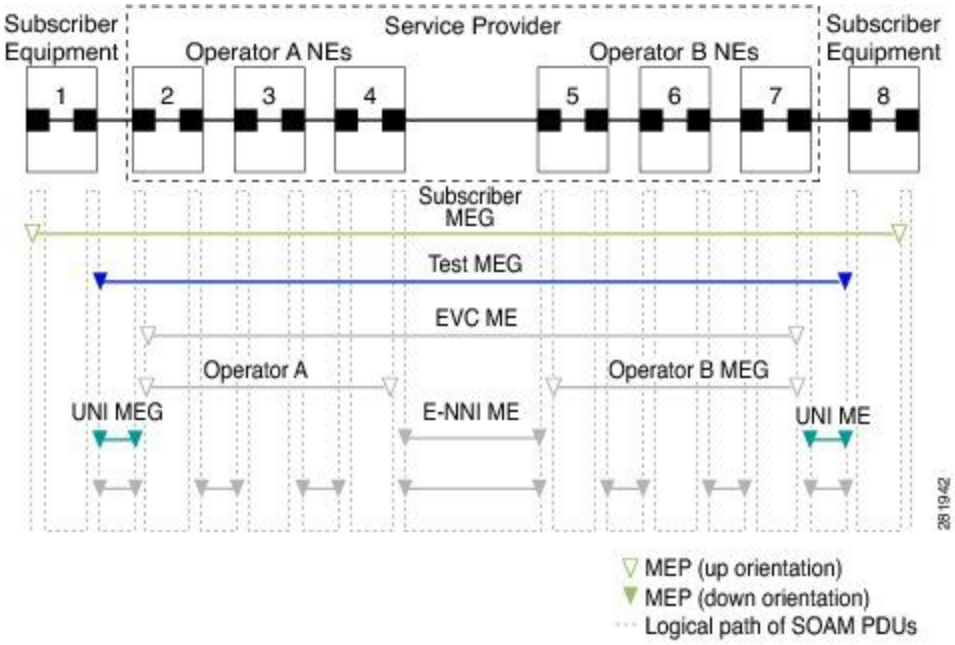

Fig 3.5.1 example of implementation

This end to end connection comes with four administrative domains:

A subscriber domain from system 1 to 8.
A service provider domain from system 2 to 7.
Two operator domains as from system 2 to 4 and 5 to 7 respectively.

Normally, the following entities are defined:

Subscriber MEG: a MEG defined amongst endpoints within the subscriber domain. This MEG has the highest MEG level and should be transparent to elements within provider and operator domains.

EVC ME: the EVC ME is defined by endpoints within the boundaries of the provider domain. It starts from an UNI and ends at another one; therefore, the performance of this ME can represent the service experience of subscriber provided by the provider.

Operator MEG: a MEG defined by endpoints within the boundaries of the operator domain. It has the lowest MEG level across domains and usually only manages the network within its domain, which is a portion of the provider network.

E-NNI ME: the ME connecting two operator MEGs within a provider domain.

UNI MEG: the MEG connecting the subscriber domain with provider domain.

In this example, periodic ETH-CC checking is performed on subscriber MEG and EVC MEG for fault detection. The endpoints of EVC MEG will also propagate ETH-BN information into subscriber domain to allow client MEPs make traffic adjustments for this service network.

When a server layer defect is detected by ETH-CC of one MEP, it will use ETH-RDI to inform other peers of this defect and MEPs on the boundaries of will send periodic ETH-AIS signals to suppress client alarms. If a client MEP detects a defect, alarms will be sent from the client towards provider edge devices; when a MEP receives a valid alarm from client, it will use ETH-CSF to inform other peers of this defect condition.

After the defect condition is detected, there are several means available to locate it. ETH-LB can be used to test connectivity between two MEPs and ETH-LT can be used to further locate the problem. If more tests are needed to be performed, MEPs can use ETH-Test in-service function to conduct various tests across the network. This can be combined with ETH-LM and ETH-DM to get the frame loss and delay of one specific connection.

If out-of-service tests are needed, then MEPs on the boundaries of the domain will send out periodic ETH-LCK signals to the subscriber domain to indicate a service interruption. Then MEPs can use ETH-Test to perform tests needed.

If a MEP foresees a interruption of ETH-CC functioning which will not affect service data, it can use ETH-ED to inform peer MEPs of this situation.

4. Conclusion

Overall, TWAMP introduces a logical model for two-way delay measurement, and Y.1731 defines a detailed implementation of OAM framework for both fault management and performance monitoring.

Comparing to each other, TWAMP is featured for its easy adoption and extension as well as security and integrity considerations, but it only covers the delay measurement part of performance monitoring. In the other hand, Y.1731 covers most of the areas of OAM with detailed functioning mechanisms. However, there are still some defects in them.

In my opinion, since TWAMP does not require clock synchronization on both sides, the Start-Time field within the Request-TW-Session would be inaccurate and should be changed from a timestamp to the period of time delayed for the test session to start.

For Y.1731 in synthetic loss measurement, the TxFCb field from the SLR PDU contains the counter value, according to the protocol, at the time of SLR frame transmission. In this way, the value might be inaccurate, since subsequent frames could arrive and change the value of the reception counter before the SLR frame is sent. The recommendation could be using the value of RxFC1 counter at the time of the SLM reception for this field.

Moreover, ETH-SLM in Y.1731 uses the RxFC1 counter value at the beginning of the measurement period to calculate the frame loss, while other elements in the formula are values at the time of the transmission or reception of synthetic frames. This will cause the inaccuracy of the result, and it should use the value of RxFC1 at the time of reception of the fist ETH-SLM frame instead.

5. Acknowledgement

I want to thank my supervisor Mr. Noonari Juned for giving guidance and patience during the project, and thank Prof. Mike MacGregor and Mrs. Sharon Gannon for providing me this opportunity to finish my project. Thank you very much!

6. Reference

1. [RFC2104] H. Krawczyk, M. Bellare, R. Canetti, “HMAC: Keyed-Hashing for Message Authentication”, February 1997.
2. [RFC2898] B. Kaliski, “PKCS #5: Password-Based Cryptography Specification Version 2.0”, September 2000.
3. [RFC4656] S. Shalunov, B. Teitelbaum, A. Karp, J. Boote, M. Zekauskas, “A One-way Active Measurement Protocol (OWAMP)”, September 2006.
4. [RFC5198] J. Klensin, M. Padlipsky, “Unicode Format for Network Interchange”, March 2008.
5. [RFC5357] K. Hedayat, R. Krzanowski, A. Morton, K. Yum, J. Babiarz, “A Two-Way Active Measurement Protocol (TWAMP)”, October 2008.
6. [Y.1730] Recommendation ITU-T Y.1730 (2004), Requirements for OAM functions in Ethernet-based networks and Ethernet services.
7. [Y.1731] Recommendation ITU-T Y.1731 (2015), Operation, administration and maintenance (OAM) functions and mechanisms for Ethernet-based networks.
8. [G.8001] Recommendation ITU-T G.8001 (2013), Terms and definitions for Ethernet frames over transport.
9. [G.8010] Recommendation ITU-T G.8010 (2004), Architecture of Ethernet layer networks.