Internet Draft

Anwar Siddiqui
Avaya Inc.
Dan Romascanu
Avaya Inc.
Eugene Golovinsky
BMC Software
22 Oct 2002

Real-time Application Quality of Service Monitoring (RAQMON) Protocol Data Unit (PDU)

<draft-siddiqui-rmonmib-raqmon-pdu-00.txt>

Status of this Memo

Abstract

This memo defines a common protocol data unit (PDU) used between RAQMON Data Source (RDS) and RAQMON Report Collector (RRC) to report a QOS statistics using RTCP and SNMP as Transport Protocol.

The original RAQMON draft [SIDDIQUI3] was split into 3 parts to identify the RAQMON Framework, RAQMON QOS PDU and RAQMON MIB.

This memo defined RAQMON QOS Protocol Data Unit (PDU). This memo also outlines mechanisms to use Real Time Transport Control Protocol (RTCP) and Simple Network Management Protocol (SNMP) to transport

these PDUs between RAQMON Data Source (RDS) and RAQMON Report Collector (RRC) as outlined in RAQMON Charter of the RMON Workgroup.

The memo [SIDDIQUI2] defines a Real-Time Application QOS Monitoring (RAQMON) Framework that extends the RMON Framework to allow Real-time Application QoS information as outlined by RAQMON Charter of the RMON Workgroup.

The memo [SIDDIQUI1] defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. The document proposes an extension to the Remote Monitoring MIB [RFC2819] to accommodate RAQMON solution.

Distribution of this memo is unlimited.

Table of Contents

1. Introduction

This memo defines a common protocol data unit (PDU) used between RAQMON Data Source (RDS) and RAQMON Report Collector (RRC) to report a QOS statistics using RTCP and SNMP as a transport protocol as outlined in RAQMON Charter of the RMON Workgroup.

The original RAQMON draft [SIDDIQUI3] was split into 3 parts to identify the RAQMON framework, RAQMON PDU and RAQMON MIB. This memo takes the portion of [SIDDIQUI3] that defined RAQMON QOS PDU and describes how various PDUs can be transported over existing Application level transport protocol like Real Time Communication protocol (RTCP) and Simple Network Management Protocol (SNMP) to transport statistics between RDS and RRC.

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 [RFC2119].

2. RAQMON Framework Overview

As outlined in [SIDDIQUI2] the RAQMON framework is based on three entities:

- RAQMON Data Source (RDS) - RAQMON Report Collector (RRC) - RAQMON MIB Structure

Figure 1 below shows various interfaces in RAQMON Framework. This draft specifies the PDUs to be transported between RDS and RRC (i.e. Interface 4 in Figure 1) using RTCP and SNMP as a transport protocol.

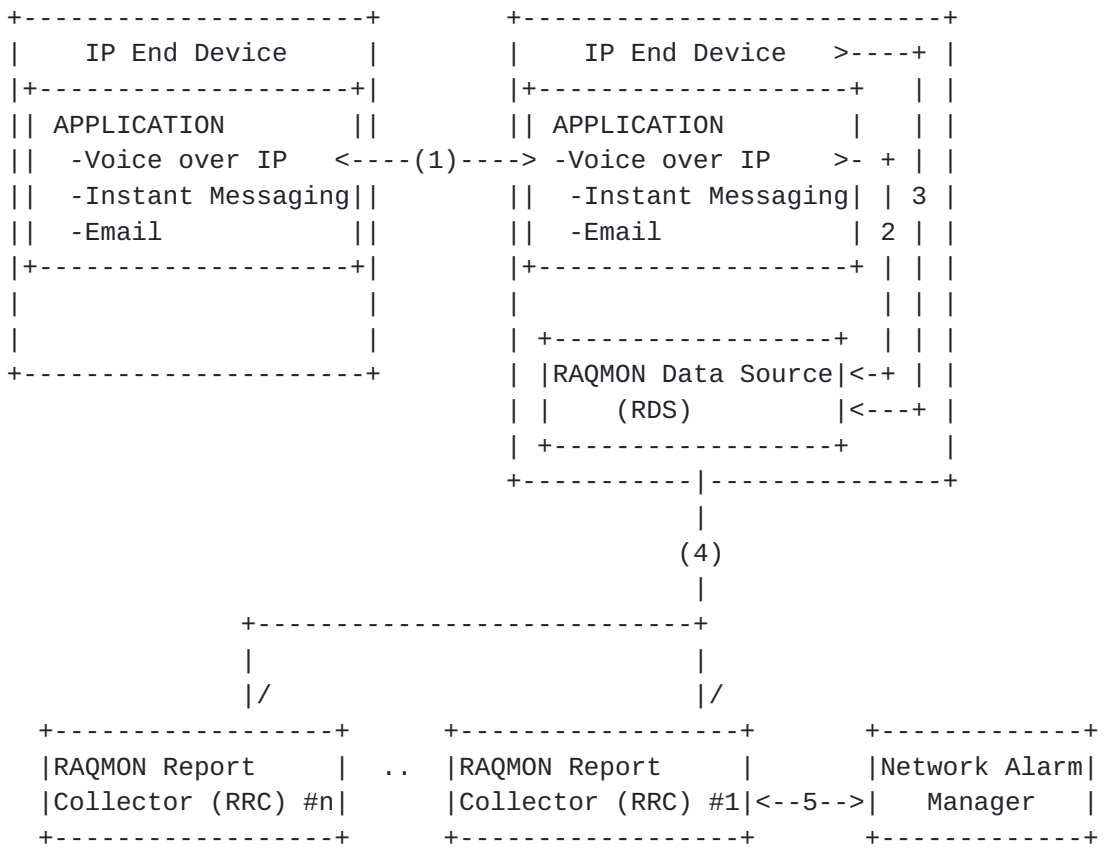


Figure 1 - RAQMON Framework.

(1) Communication Session between IP end devices/apps affected by underlying transport network

(2) Context-Sensitive transport network Specific Metrics

(3) Device State Specific Metrics

(4) RAQMON PDU transmitted over this interface (IP Address, port)

(5) RAQMON MIB sent within SNMP notifications.

3. RAQMON Protocol Data Unit (PDU) Design Overview:

This document continues the architecture created in the RMON MIB [RFC2819] by providing analysis of application performance as experienced by end-users on a specific IP end point and correlating such performance statistics to its underlying transport network characteristics. This memo is written with following assumptions:

+ All IP end points and applications are producers and consumers of IP Traffic.

+ The design of the RAQMON QOS PDUs are such that, it can be used by many Real-Time Applications like Voice over IP, Fax over IP, Video over IP, IP Short Messaging Services (SMS), Instant Messaging, Email, chats, ftp/tftp based downloads, e-business style transactions, web access etc.

+ RAQMON PDUs are agnostic to the underlying measurement methodology used to quantify a PDU parameter.

RAQMON PDUs offer an entry (a.k.a. "Name") to be filled in by application specific software which with a specific "value". Since RAQMON PDUs are common data formats commonly understood by RDS and RRC to exchange RAQMON Statistics (i.e. "Name" and "Value" pair), measurement methodologies are out of the scope of RAQMON specification. It is also out of the scope of PDU specification to validate specific measurement methodology used to gather a "value".

However set of "Name" entries specified in RAQMON PDU in this draft can be filled in with a "value" using IPPM WG recommended measurement methodologies.

+ In order to facilitate complete End-to-End view and to portray end user experience, the RAQMON PDUs SHOULD be able to carry statistics that relates to

i. "User, Application, Session" specific parameters ii. "IP end device" specific parameters during a session iii. "Transport network"

specific parameter during a session

User experience of an application running on a specific IP end point has lot to do with the type of application an user is running, local end device resources available as well as the underlying transport network capabilities.

+ RAQMON PDU "Names" are selectable by the RDS and Application implementation.

End-to-End QOS view is sensitive to application type, device and transport network. Though RAQMON PDUs are capable of carrying various pre-specified parameters but, it is expected that proposed PDUs MUST provide options to select a sub-set of those parameters from the metrics definition list, to fit the needs of the application-context. The Application implementer controlling the RDS will be responsible for choosing a set of parameters, as "monitoring context" is application specific.

For example an IP Soft Phone application running on a PC probably be willing to report "Jitter" to RRC however an Email Application running on the same host may not use the "Jitter" parameter to report to RRC as Jitter is deemed to be not so critical for Email Application.

+ List of "Names" used by the RAQMON PDU MUST be extensible to accommodate Application and vendor specific implementations.

+ Monitored statistics is reported by the RAQMON Data Source (RDS) at will.

Transmission timing and frequency of RAQMON PDUs will be completely controlled by the RDSs to provide ease of management and administration.

Though monitoring is a useful function but there are various operation scenarios where monitoring could be expensive and degrade the QOS of an application. There are also restrictions imposed on end devices based on the administrative domains. For example, an Enterprise IP Phone user is managed by the enterprise Telecom manager, but the Service Level Agreements is monitored by the Enterprise and ISP IT Managers. In such an environment, IP Phones may be required to report QOS Problems to various administrative authorities restricted by the administration domain policy. A RDS Driven reporting mechanism allows enough flexibility to accommodate various administrative constraints.

+ Quality of service parameters of each communication session should

be captured and stored completely.

A communication session may consist of one or more combinations of transaction-oriented, throughput-oriented, or streaming-oriented operations. For example, the quality-of-service definition of a Video over IP call using Video Phones involves

- Caller Video Phone signaling for call setup that includes a transaction with a session processing server which locates/connects the callee using a protocols like SIP, H.323 or MGCP.

- Eventually the video phone source/sinks media streams between two IP end points using RTP as a result of successful session setup transaction

In this particular application scenario, the Session Set up timing is as critical as the End-to-End Delay per packet of media streams. The RAQMON PDUs should provide a capability to capture such session specific data.

RAQMON draft would use the following definitions of transactions as defined in the APM MIB [WALDBUSSER]:

Transaction-Oriented: These transactions have a fairly constant workload to perform for all transactions. The responsiveness metric for transaction-oriented applications is application response time, the elapsed time between the user's request for service (e.g. pushing the submit button or pressing DTMF in IP Phones) and the completion of the request (e.g. displaying the results or getting a ring back).

Throughput-Oriented: These transactions have widely varying workloads based on the amount of data requested. The metric for throughput-oriented applications are expressed in is Kilobits per second (Kbps) or Mega bits per second (Mbps).

Streaming-Oriented: These transactions deliver data at a constant metered rate of speed regardless of excess capacity in the networking and computing infrastructure. However, when the infrastructures cannot deliver data at this speed, interruption of service or degradation of service can result.

+ A report on a communication session between users should capture the entire session by keeping records of all the sub-sessions performed within that session.

A generic communication session between two users can be modeled as multiple sub-sessions within a communication session. For example a video call between two users would capture Quality of Service

parameters of a session for Audio, Video and Data separately but within one compound report as it reflects the true nature of the communication session. It is easier for an end device to correlate between these sub-sessions and report the End-to-End QOS parameters of that session in a compound report.

+ The monitoring functionality must run in real-time during each communication session and consume very minimal device resources.

Many of the IP end points that runs applications like Voice over IP, Fax over IP, Video over IP, Short Messaging Services (SMS), Instant Messaging, Email, chats, ftp/tftp based downloads, e-business style transactions, web access are embedded devices with resources constraints.

Monitoring of these devices and applications is performed for all communication session as QOS of each session is dependent on the time when monitoring was performed.

+ RAQMON Framework requires a simple, easy to understand and simple to implement metrics definition.

Metrics definitions need to be simple and intuitive to Application Service Providers, IT Managers, network operators, equipment vendors etc.

+ RAQMON PDUs design should be embedded device friendly.

The applications covered under the RAQMON Charter have become such a commodity in our everyday lives that there are lots of simple embedded smart devices being developed by various vendors at an enormous rate. Application Service Providers, Network Service Providers, Enterprise operators, IT Managers etc. have an inherent need to gather QOS Reports of these devices and applications to manage there networks and services. It is the objective of this draft to deliver a simple but easy to deploy monitoring solution.

+ RDS and RRC will be using either RTCP or SNMP to transport RAQMON PDUs.

4. A Simple Metrics

The objectives set in the previous section dictate that the RAQMON framework ought to provide a simple metrics definition. It is an extremely challenging task to define "appropriate metrics" as metrics are context-sensitive. However one can also notice that there are enough commonalities between the various QOS parameters associated to various applications such that the task of defining a "simple

metrics" is feasible. This document defines a simple metric that in essence captures the performance and associated quality-of-service parameters of a communication session. RAQMON framework also provides a mechanism to add and drop various parameters to this metrics as defined in Table 1 below to accommodate application context sensitivity:

1. Data Source Name (DN)

2. Receiver Name (RN)

3. Data Source Address (DA)

4. Receiver Address (RA)

5. Data Source Device Port used

6. Receiver Device Port used

7. Session Setup Date/Time

8. Session Setup delay

9. Session duration

10. Session Setup Status

11. End-to-End Delay

12. Inter Arrival Jitter

13. Total number of Packets Received

14. Total number of Packets Sent

15. Total number of Octets Received

16. Total number of Octets Sent

17. Cumulative Packet Loss

18. Packet Loss in Fraction

19. Source Payload Type

20. Receiver Payload Type

21. Source Layer 2 Priority

22. Destination Layer 2 Priority

23. Source Layer 3 Priority

24. Destination Layer 3 Priority

25. CPU utilization in Fraction

26. Memory utilization in Fraction

27. Application Name/version

28. RAQMON Optional Flag (ROF)

Table 1: RAQMON Metrics Definition

Various parameters listed in table 1 are defined below. The definition presented here is meant to provide guidance to implementers. No claim is made that the definitions presented here are appropriate for a particular application need.

Data Source Name (DN): The DN item could be of various formats as needed by the application. Few instances of DN could be but not restricting to

* "user@host", or "host" if a user name is not available as on single-user systems. For both formats, "host" is either the fully qualified domain name of the host from which the payload originates, formatted according to the rules specified in [RFC1034], [RFC1035] and Section 2.1 of [RFC1123]; The DN value is expected to remain constant for the duration of a session. Examples are "big-guy@ip-phone.bigcompany.com" or "big-guy@135.8.45.178" for a multi-user system. On a system with no user name, examples would be "ip-phone4630.bigcompany.com". It is recommended that the standard host's numeric address not be reported via DN parameter as Data Source Address (DA) parameter is used for that purpose.

* Another instance of a DN could a valid E.164 phone number, a SIP URI or any other form of telephone or pager numbers. It is recommended that the phone number should be formatted with the plus sign replacing the international access code. For example, "+88 02 123 45678" for a number in Bangladesh.

It is expected that a Data Source Name (DN) will remain constant within a communication session.

Receiver Name (RN): Same as Data Source Name (DN).

Data Source Address (DA): Data Source Address (DA) parameter should be represented as the standard ASCII representation of the host's numeric address. This could be an IPv4 Address, IPv6 Address, network address assignments such as the Net-10 assignment proposed in [RFC1597] or any other form of numeric address represented in ASCII.

It is expected that a Data Source Name (DN) would remain constant within a communication session.

DN and DA are intended to give the application writers an opportunity to uniquely identify a record associated to a session. However application writers should be aware that private network address assignments such as the Net-10 assignment may create network addresses that are not globally unique. To handle this case, the burden is on the application either by converting private addresses to public addresses if necessary to keep private addresses from being exposed or by creating an application specific extension.

Receiver Address (RA): Same as Data Source Address

Data Source Device Ports used: This parameter is used to indicate the port used for a particular session or sub-session used for communication. Example of port includes TCP Port, UDP Port, RTP Port etc. It is not expected that a Data Source Device Ports would remain constant within a communication session.

Receiver Device Ports used: Same as Data Source Device Ports used.

Session Setup Date/Time Indicates the wallclock time when the RAQMON packet was sent so that it may be used by the RRC to store Date/Time. Wallclock time (absolute time) is represented using the timestamp format of the Network Time Protocol (NTP), which is in seconds relative to 0h UTC on 1 January 1900 [RFC1305].

Session Setup delay: Session setup delay indicates the duration of time required by a network communication controller to set a media path between the communicating entities or the end devices. For example in VoIP systems a session setup time can be measured as the last DTMF button pushed to the first ring back tone that indicates that the far end is ringing. However as these definitions are very specific to the type of system used and implementation details of such system, no claim is made about the definition presented here are appropriate for a particular application need and left upon the implementers to define.

Session duration: This parameter describes how long a session or a sub-session lasted.

Session Setup Status: This parameter is intended to report status of a session in order to support applications those need to display status in realtime. For example a debugging tool that captures the status of a call setup as soon as a call is established or a tool that captures why a session failed or how many RSVP session failed etc.

End-to-End Delay: End-to-End delay is a key parameter for Application QOS Monitoring. Some applications do not perform well (or at all) if end-to-end delay between hosts is large relative to some threshold value. Erratic variation in delay makes it difficult (or impossible) to support many real-time applications like Voice over IP, Video over IP, Fax over IP etc.

There are many measurement methodologies available to fill this parameter but this parameter is intended to capture the End-to-End delay as observed by the IP devices at the application layer pertaining to a specific operation environment. While appropriate, it is recommended that specific application layer delays like play out delay, packet sequencing delays, coding, decoding delays be added to transport network delay to report End-to-End delay under RAQMON Framework.

End-to-End delay of underlying transport network can be measured using various methodologies as described in [RFC2681], [RFC2679], [RFC1889] depending on the application needs and left upon the implementers based on there application need.

Inter-arrival Jitter: Inter-arrival jitter field provides a short-term measure of congestion. The definition of Jitter is context sensitive and measurement specific. Measurement of inter-arrival Jitter is beyond the scope of this document. The jitter measure indicates congestion before it leads to packet loss. Inter-arrival jitter of underlying transport network can be measured using various methodologies and left upon the implementers based on there application need. VoIP Systems can readily acquire Inter-arrival Jitter calculations from RTCP measurements as described in [RFC1889].

Total number of Packets Received: The total number of packets received by the data source since starting transmission up until the time this RAQMON packet was generated.

Total number of Packets Sent: Similar to total number of packets received.

Total number of Octets Received: The total number of payload octets received in packets by the sender since starting transmission up until the time this RAQMON packet was generated.

Total number of Octets Sent: Similar to total number of octets received.

Cumulative Packet Loss: Packet loss tracks persistent congestion while the jitter measure tracks transient congestion. Since the interarrival jitter field is only a snapshot of the jitter at the time of a report, packet loss indicates the network environment as well as local device losses over time. Packet loss of underlying transport network can be measured using various methodologies e.g. as described in [RFC2680], [RFC1889] and local device level packet losses ought to be captured by the local device specific algorithms. Measurement methodologies are left upon the implementers based on there application need.

Packet loss in Fraction: Same as Packets loss but expressed in percentage

Source Payload Type: Defines payload formats (e.g. media encodings) as sent by the data source. e.g. ITU G.711-(law, ITU G.729B, H.263, MPEG-2, ASCII etc. This document follows the same payload type constants as defined in [RFC1890].

Destination Payload Type: Similar to Source Payload Type.

Source Layer 2 Priority: Many devices use Layer 2 technologies to prioritize certain type of traffic in the Local Area Network environment. For example the 1998 Edition of IEEE 802.1D [IEEE802.1D] "Media Access Control" Bridges contains expedited traffic capabilities to support transmission of time critical information and many devices use the standard to mark Ethernet frames according to IEEE 802.1p standard. Details on these can be found in IEEE 802.1Q "Virtual Bridged LAN" specifications. 802.1p has been Incorporated into ISO/IEC 15802-3 1998 [IEEE802.1Q]. Source Layer 2 RAQMON field indicates Layer 2 values used by the Data Source to prioritize these packets in the Local Area Networks environment.

Source Layer 3 Priority: Various Layer 3 technologies are in place to prioritize certain type of traffic in the internet. For example traditional IP Precedence [RFC791], Type Of Service (TOS) [RFC1349],[RFC1812] or more recent technologies like Differentiated Service [RFC2474][RFC2475] is achieved by using the TOS octet in IPv4 and the Traffic class Octet in IPv6 are used to prioritize traffic. Source Layer 3 RAQMON field indicates appropriate Layer 3 values used by the Data Source to prioritize these packets.

Destination Layer 2 Priority: Same as Source Layer 2 Priority.

Destination Layer 3 Priority: Same as Source Layer 3 Priority.

CPU utilization in Fraction: This parameter captures the IP Device CPU usage rate to indicate current state of the local IP Device resource which has a very critical implications on QOS implications of an end device. e.g. x % CPU busy averaged over session duration.

Memory utilization in Fraction: This parameter captures the IP Device Memory usage rate to indicate current state of the local IP Device resource which has a very critical implications on QOS implications of an end device. e.g. y % memory utilized over session duration.

Application Name/version Application Name/version parameter gives the name and possibly version of the application associated to that session or sub-session, e.g., "XYZ VoIP Agent 1.2". This information may be useful for scenarios where end device is running multiple applications with various priorities and could be very handy for debugging purposes.

RAQMON Optional Flag (ROF): These flags are open to various vendors to be used for application specific bit level signaling. For example RDS can report various numeric status code to RRCs using these bits. For example, the end devices that support RSVP to setup a communication session would be successful in acquiring RSVP reservation in one direction but not the other. A specific 8-bit failure code can be used to indicate each failure code. One could also use these bits to indicate RAQMON packet sequence number. These 8-bit Optional Flags are interpreted by the application, not by the RRC and usage of these left at the application developer's discretion.

4.1 Measurement Methodology

It is not the intent of this document to recommend a methodology to measure any of the QOS parameters defined in table 1. Measurement algorithms are left upon the implementers and equipment vendors to choose. There are many different measurement methodologies available for measuring application performance (e.g., probe-based, client-based, synthetic-transaction, etc.). This specification does not mandate a particular methodology - it is open to any that meet the minimum requirements. Conformance to this specification requires that the collected data match the semantics described herein.

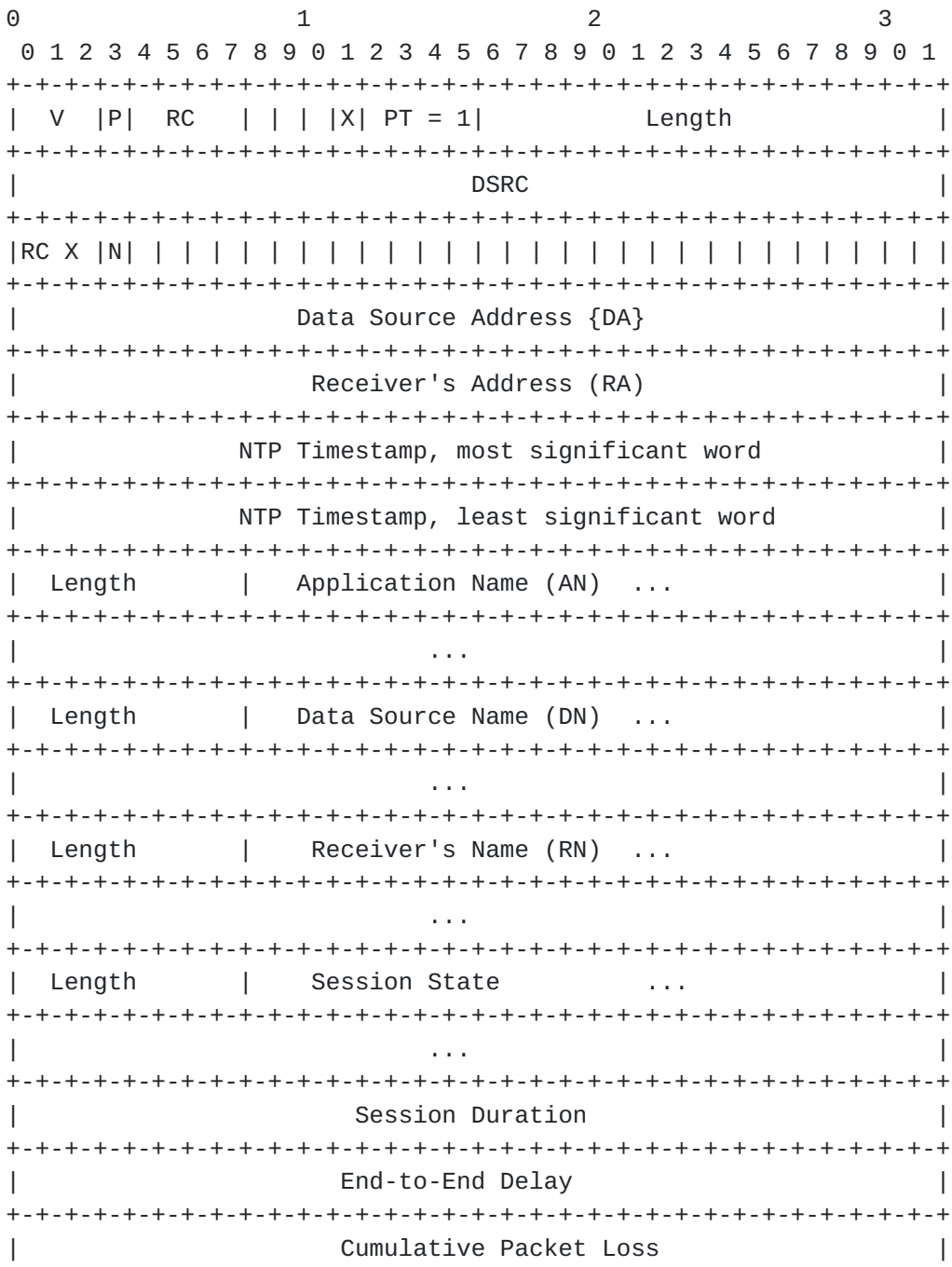
5. RAQMON PDU Format:

There are 2 types of RAQMON PDUs used by the RDS to report various QOS parameters to RRC.

BASIC: For reporting monitored data from an RDS to RRC which includes QOS parameters defined in table 1. BASIC Packets are marked as PAT = 1

APP: These APP Packets can define Application specific PDUs. APP PDUs are marked as PAT = 4

Following is various RAQMON PDU formats:



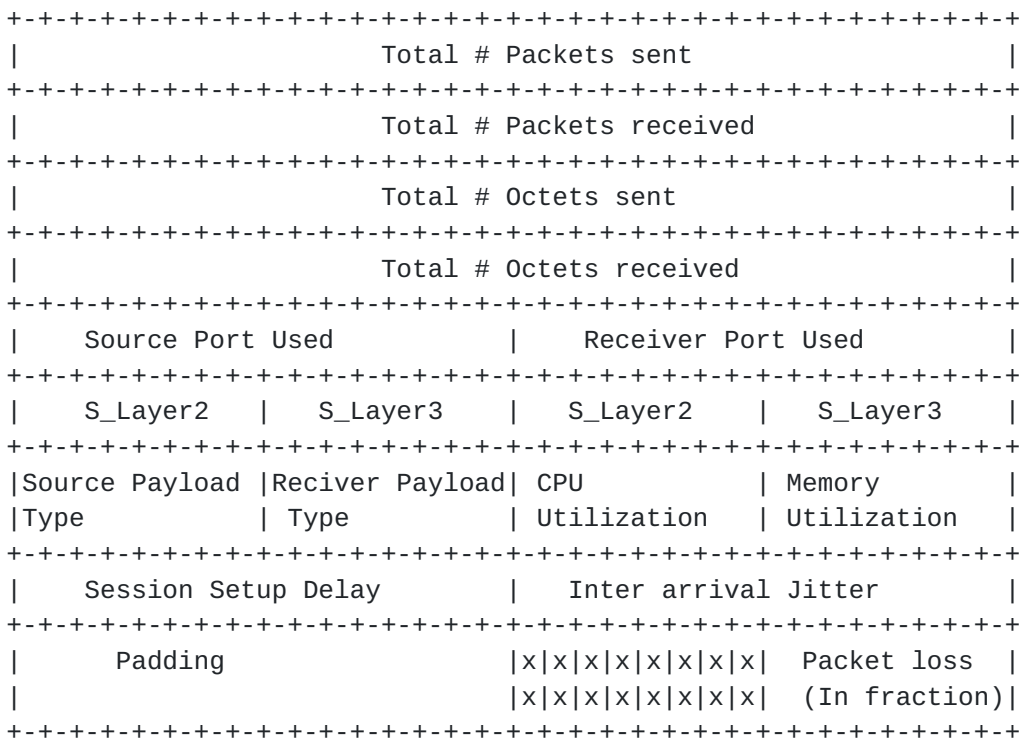


Figure 2 - Basic Protcol Data Unit

5.1 BASIC Protocol Data Unit (PDU):

version (V) : 3 bits - Identifies the version of RAQMON. This version is 1.

padding (P): 1 bit - If the padding bit is set, this RAQMON packet contains some additional padding octets at the end which are not part of the monitoring information. The last octet of the padding is a count of how many padding octets should be ignored. Padding may be needed by some applications as reporting is based on the intent of RDS to report certain parameters.

record count (RC): 4 bits - Total number of records contained in this packet. A value of zero is valid but useless.

reserved bits: 3 bits - reserved for future extensions to the RAQMON Packet.

IPversion Flag: 1 bit - While set to 1, IP Version Flag indicates that IP addresses are IP version 6 compatible.

Packet Type (PAT): 4 bits - This indicates the type of RAQMON packet being sent. There are 2 types of RAQMON Packets. BASIC Packets (PAT = 1) and APP Packet (PAT =4).

length: 16 bits - The length of this RAQMON packet in 32-bit words minus one which includes the header and any padding.

DSRC: 32 bits - Data Source identifier represents a unique session descriptor that points to a specific communication session between communicating entities. Uniqueness of DSRC is valid only within a session. DSRC values should be randomly generated using vendor chosen algorithms. It is not sufficient to obtain a DSRC simply by calling random() without carefully initializing the state. It is beyond the scope of this document define an algorithm to generate DSRC. However one could very easily use an algorithm like the one defined in Appendix A.6 in [17]. Depending on the choice of algorithm, there is a finite probability that two DSRCS from two different RDSs may be same. To further reduce the probability that two RDSs pick the same DSRC, it is recommended that an RRC or an application use Data Source Address (DA) and Data Source Name (DN) in conjunction with a DSRC value to reduce that probability drastically.

Each RAQMON packet consists of a DSRC followed by RC_n and RAQMON flags to indicate presence of appropriate RAQMON parameters as defined in table 1.

RC_n: 4 bits - Record Count number to which the information in this record pertains. Record Count number indicates a sub-session within a communication session. A value of zero is a valid record number. Maximum number of records that can be described in one RAQMON Packet is 16 (i.e. 0000 - 1111).

RAQMON Parameter Presence Flags (RPPF): 28 bits

Each of these flags while set represent that this RAQMON packet contains corresponding parameters as specified in table 2

Sequence Number	Presence/Absence of corresponding Parameter within this RAQMON packet
1	Data Source Address (DA)

2	Receiver Address (RA)
3	NTP Timestamp
4	Application Name
5	Data Source Name (DN)
6	Receiver Name (RN)
7	Session Setup Status
8	Session Duration
9	End-to-End Delay
0	Cumulative Packet Loss
1	Total number of Packets sent
2	Total number of Packets received
3	Total number of Octets sent
4	Total number of Octets received
5	Source Port Used
6	Receiver Port Used
7	S_Layer2
8	S_Layer3
9	D_Layer2
0	D_Layer3
1	Source Payload Type
2	Receiver Payload Type
3	CPU Utilization
4	Memory Utilization
5	Session Setup Delay

6	Inter arrival Jitter
7	Packet loss (in fraction)
8	RAQMON Optional Flag (ROF)

Table 2: RAQMON Parameters and corresponding RPPF

Data Source Name: - Data Source Name field starts with an 8-bit octet count describing the length of the text and the text itself. Note that the text can be no longer than 255 octets. The text is encoded according to the UTF-2 encoding specified in Annex F of ISO standard 10646 [ISO10646],[UNICODE]. This encoding is also known as UTF-8 or UTF-FSS. It is described in "File System Safe UCS Transformation Format (FSS_UTF)", X/Open Preliminary Specification, Document Number P316 and Unicode Technical Report #4. US-ASCII is a subset of this encoding and requires no additional encoding. The presence of multi-octet encoding is indicated by setting the most significant bit of a character to a value of one. Text is not null terminated because some multi-octet encoding include null octets. Data Source Name is terminated by one or more null octets, the first of which is interpreted as to denote the end of the string and the remainder as needed to pad until the next 32-bit boundary. Since the Data Source Name is expected to remain constant for the duration of the session, it is recommended that RDS report such field only once within a communication session to ensure efficient usage of network and system resources.

Receiver Name: - Same as Data Source Name. Data Source Name and Receiver's Name are contiguous, i.e., items are not individually padded to a 32-bit boundary.

Data Source Address: 32 bits - The standard ASCII representation of the end device's numeric address on the interface used for the communication session. The standard ASCII representation of an IP Version 4 address is "dotted decimal", also known as dotted quad. Other address types are expected to have ASCII representations that are mutually unique. 135.8.45.178 is an example of a valid Data Source Address. Since the Data Source Address is expected to remain constant for the duration of the session, it is recommended that RDS report such field only once within a communication session to ensure efficient usage of network and system resources.

Issue: IP addresses, TCP/UDP ports information should be removed (NAT un-friendly). One of the way to avoid this problem is to use Application Layer Gateways (ALGs) to fill out IP Addresses on RDS's behalf.

Receiver Address: 32 bits - Same as Data Source Address

Application Name: - Application Name field starts with an 8-bit octet count describing the length of the text and the text itself. Application name field has same format as Data Source Name. This is a text string giving the name and possibly version of the application associated to that session, e.g., "XYZ VoIP Agent 1.2". This information may be useful for debugging purposes and is similar to the Mailer or Mail-System-Version SMTP headers. Since the Application Name is expected to remain constant for the duration of the session, it is recommended that RDS report such field only once within a communication session to ensure efficient usage of network and system resources.

NTP timestamp: 64 bits - Indicates the wallclock time when the RAQMON packet was sent so that it may be used by the RRC to store Date/Time. A Data Source that has no notion of wallclock or time may set the NTP timestamp to zero. However that will waste 32 bits in the packet. An RDS should set the appropriate RAQMON flag to 0 to avoid such waste. Since NTP time stamp is intended to provide Date/Time of a session, it is recommended that the NTP Timestamp be used only in the first RAQMON packet to use network resources efficiently. However such a recommendation is context sensitive and should be enforced as deemed necessary by each application environment.

The full resolution NTP timestamp is a 64-bit unsigned fixed-point number with the integer part in the first 32 bits and the fractional part in the last 32 bits. In some fields where a more compact representation is appropriate, only the middle 32 bits are used; that is, the low 16 bits of the integer part and the high 16 bits of the fractional part. The high 16 bits of the integer part must be determined independently.

Session Setup Status: - Session State field starts with an 8-bit octet count describing the length of the text and the text itself. This field is used to describe appropriate communication session states e.g. Call Established successfully, RSVP reservation failed etc.

Session Duration: 32 bits - Session Duration is an unsigned Integer expressed in the order of seconds.

End-to-End Delay: 32 bits - End-to-End Delay is an unsigned Integer expressed in the order of milliseconds.

Cumulative Packet Loss: 32 bits - The total number of packets from session RC_n that have been lost while this RAQMON packet was generated. This number is defined to be the number of packets

expected less the number of packets actually received.

Total number of Packets sent: 32 bits - The total number of packets transmitted within a communication session by the sender since starting transmission up until the time this RAQMON packet was generated. This counter is reset if the DSRC identifier is changed as it indicates a different session.

Total number of Packets received: 32 bits - The total number of packets transmitted within a communication session by the receiver since starting transmission up until the time this RAQMON packet was generated. This counter is reset if the DSRC identifier is changed as it indicates a different session.

Total number of Octets sent: 32 bits - The total number of payload octets (i.e., not including header or padding) transmitted in packets by the sender within a communication session since starting transmission up until the time this RAQMON packet was generated. This counter is reset if the DSRC identifier is changed as it indicates a different session.

Total number of Octets received: 32 bits - The total number of payload octets (i.e., not including header or padding) transmitted in packets by the receiver within a communication session since starting transmission up until the time this RAQMON packet was generated. This counter is reset if the DSRC identifier is changed as it indicates a different session.

Source Port Used: 16 bits - Port Number used by the Data Source as used by the application while this RAQMON Packet was generated.

Receiver Port Used: 16 bits - Same as Source Port Used

S_Layer2: 8 bits - Source Layer 2 priorities used to send packets to the receiver by this data source during this communication session. For example priority bits associated to IEEE 802.1p values for appropriate priorities. For example priority bits associated to IEEE 802.1p tags value of 5 reported via S_Layer2 parameter would indicate Video over IP from this data source is prioritized by some Layer 2 switch.

S_Layer3: 8 bits - Layer 3 priorities used to send packets to the receiver by this data source during this communication session. For example priority bits associated to IP Precedence (i.e. 101XXXXX) or DiffServ PHB values (i.e EF, AF41) etc reported via S_Layer3 parameter would indicate whether applications from this data source is prioritized by some Layer 3 switch or not.

D_Layer2: 8 bits - Layer 2 priorities used by the receiver to send packets to the data source during this communication session if the Data Source can learn such information.

D_Layer3: 8 bits - Layer 3 priorities used by the receiver to send packets to the data source during this communication session if the Data Source can learn such information.

Source Payload Type: 8 bit - This document follows definition of Payload Type (PT) as definition is in [RFC1890]. This 8 bit fields specify the type of audio, video or data media used to send packets to the receiver by this data source during a communication session. Table 3 indicates a small list of various Payload types as defined in [RFC1890] cited here for informational purposes. As this table indicates, if an application ought to indicate that the Source Pay Load Type used for a session were PCMA, Source Payload Field of the BASIC RAQMON packet ought to be 8.

PT	encoding name	audio/video (A/V)	clock rate (Hz)	channels (audio)
0	PCMU	A	8000	1
1	1016	A	8000	1
2	G721	A	8000	1
3	GSM	A	8000	1
4	unassigned	A	8000	1
5	DVI4	A	8000	1
6	DVI4	A	16000	1
7	LPC	A	8000	1
8	PCMA	A	8000	1
9	G722	A	8000	1
10	L16	A	44100	2
11	L16	A	44100	1
12	unassigned	A		
13	unassigned	A		
14	MPA	A	90000	(see text)
15	G728	A	8000	1
16--23	unassigned	A		
24	unassigned	V		
25	CelB	V	90000	
26	JPEG	V	90000	
27	unassigned	V		
28	nv	V	90000	
29	unassigned	V		
30	unassigned	V		
31	H261	V	90000	
32	MPV	V	90000	

33	MP2T	AV	90000	
34--71	unassigned	?		
72--76	reserved	N/A	N/A	N/A
77--95	unassigned	?		
96--127	dynamic	?		

Table 3: Payload types (PT) for standard audio and video encodings

Please refer to [RFC1890] for various other Audio, Video and Data related payload types.

CPU Utilization: 8 bits - Percentage of CPU used over a time duration.

Memory Utilization: 8 bits - Percentage of total memory over a time duration.

Session Setup Delay: 16 bits - Indicates the duration of time required by a network communication controller to set a media path between the communicating entities or the end devices. This parameter is expressed in milliseconds.

Inter-Arrival Jitter: 16 bits - An estimate of the statistical variance of packets inter-arrival time expressed in milliseconds.

Packet Loss in Fraction: 8 bits - The fraction of packets from data source lost since the previous RAQMON was dispatched, expressed as a fixed point number with the binary point at the left edge of the field. (That is equivalent to taking the integer part after multiplying the loss fraction by 256.) This fraction is defined to be the number of packets lost divided by the number of packets expected.

RAQMON Optional Flag: 8 bits - These bits are open to various vendors to be used for application specific bit level signaling. These 8-bit Optional Flags are interpreted by the application, not by the RRC and usage of these left at the application developer's discretion.

5.2 Mapping of Basic RAQMON Packet to SNMP notification.

The information carried by Basic RAQMON packet MAY be delivered by SNMP notifications. This delivery mechanism works in conjunction with RAQMON Notifications defined in [SIDDIQUI1]. As described in Section 5.1, the use of SNMP Informs is RECOEMDED. Full compliance with RFC2273 to support Command Responder/Notification Originator applications is NOT REQUIRED. This is to be left up to implementer. A RAQMON device can implement either a full SNMP agent, or a subset that sends RAQMON PDUs in a format similar to SNMP Informs. The section of the draft defines mapping mechanism of information carried

by Basic RAQMON Packet to SNMP notification PDU(s).

5.3 APP Protocol Data Unit:

The APP PDU is intended for experimental use as new applications and new features are developed, without requiring packet type value registration. APP packets with unrecognized names should be ignored. After testing and if wider use is justified, it is recommended that each APP packet be redefined without the subtype and name fields and registered with the Internet Assigned Numbers Authority (IANA).

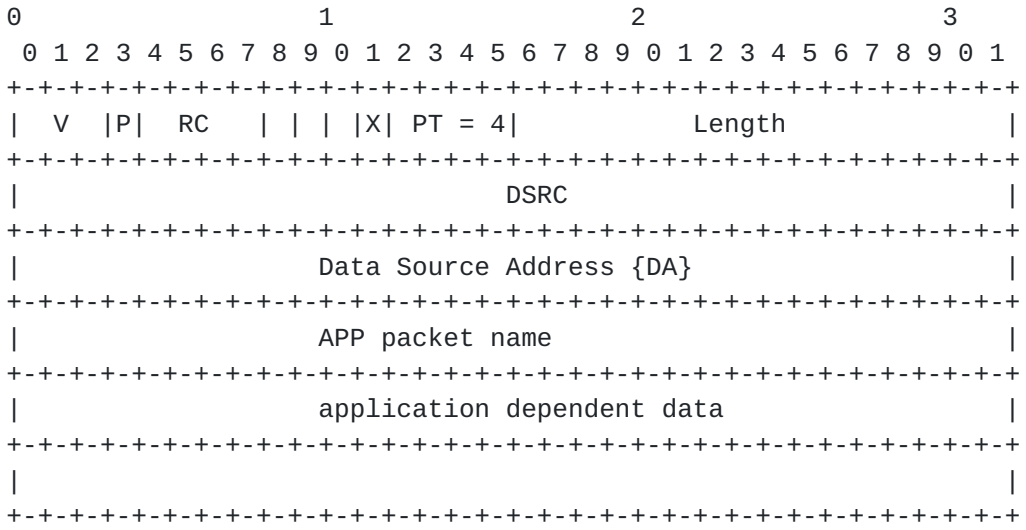


Figure 3 - RAQMON APP Protcol Data Unit

version (V), padding (P), record count (RC): As defined for BASIC Packet.

reserved bits: 3 bits - reserved for future extensions to the RAQMON Packet.

IPversion Flag: As defined for BASIC Packet.

DSRC and DA: As defined for BASIC Packet.

subtype: 4 bits - May be used as a subtype to allow a set of APP PDUs to be defined under one unique name, or for any application-dependent data.

pdu type (PAT): 4 bits - Contains the constant 4 to identify this as an RAQMON APP PDU.

name: 4 octets - A name chosen by the person defining the set of APP

PDUs to be unique with respect to other APP PDUs this application might receive. The application creator might choose to use the application name, and then coordinate the allocation of subtype values to others who want to define new packet types for the application. Alternatively, it is recommended that others choose a name based on the entity they represent, then coordinate the use of the name within that entity. The name is interpreted as a sequence of four ASCII characters, with uppercase and lowercase characters treated as distinct.

application-dependent data: variable length - Application-dependent data may or may not appear in an APP packet. It is interpreted by the application and not by the RRC itself. It must be a multiple of 32 bits long.

5.4 Byte Order, Alignment, and Time Format of RAQMON PDUs

All integer fields are carried in network byte order, that is, most significant byte (octet) first. This byte order is commonly known as big-endian. The transmission order is described in detail in [RFC791]. Unless otherwise noted, numeric constants are in decimal (base 10).

All header data is aligned to its natural length, i.e., 16-bit fields are aligned on even offsets, 32-bit fields are aligned at offsets divisible by four, etc. Octets designated as padding have the value zero.

6. Transporting RAQMON Protocol Data Units

It is an inherent objective of the RAQMON Framework to re-use existing application level transport protocols to maximize the usage of existing installations as well as to avoid transport protocol level complexities in the design process. As outlined in the RAQMON framework document that both the Real-Time Transport Control Protocol and Simple Network Management Protocol were suitable to meet the criteria of a transport protocol as outlined in the RAQMON Charter. Section 5. 1 reflects mechanisms that uses SNMP INFORM PDUs as transport protocol and section 5.2 elaborates a protocol that uses RTCP APP Packets [RFC 1889] to transport RAQMON PDUs between RDS and RRC.

6.1 SNMP INFORM PDUs as RDS/RRC Network Transport Protocol The idea is to re-use SNMP INFORM PDU. This proposal offers that:

+ RDSs implement the capability of embedding RAQMON parameters in SNMP INFORM Request and thus re-using well known SNMP mechanisms to

report RAQMON Statistics.

+ To keep the RDS realization simple and keep the protocol lightweight, the RDSs will not be REQUIRED to respond to SNMP requests like get, set, etc., as an SNMP compliant responder would.

+ If the RRC chooses to implement an SNMP manager, an SNMP INFORM Response would be sent for each associated SNMP INFORM originated by the RDS.

+ The RDS may ignore the SNMP INFORM Responses, or, if better reliability is required, will wait for the Inform response, retransmitting the original Inform PDU every M seconds until it has been sent N times.

+ The SNMP INFORM transport for RAQMON PDUs can use one of the two UDP ports assignments:

- Standard UDP port 162 used for SNMP Notifications, if full SNMP entities implementations are present in the RRC and RDS

- IANA assigned UDP port 5YYYY for RAQMON PDUs carried over SNMP, for the cases when at least one of the RRC and RDS does not support a full implementation of the SNMP entities.

The benefits of using SNMP Informs are: - Using a well-known protocol. - Privacy and authentication are covered by SNMPv3 - Limited or no need for specific RAQMON-protocol code in the RRC, as it can use an SNMP manager implementation to process Informs.

The drawback of this approach is the overhead SNMP puts on low-powered RDSs, for instance - BER encoding.

6.1.1 Encoding RAQMON PDU format within a small set of MIB items.

The RAQMON PDU defined in Section 4.1 is encapsulated in the raqmonPDUBasicPDU MIB object from the RAQMON MIB [SIDDIQUI1]. This object has a SYNTAX of an OCTET STRING variable, which encodes the content of the data fields described in figure 2. The Inform Request will contain this object.

6.1.2 SNMP Inform PDU Related Issues as applied to RAQMON

Using SNMP INFORM PDUs for RAQMON has all the advantages offered by a well known protocol like SNMP. Privacy and authentication issues related to RAQMON are "mostly" covered by SNMPv3

However there are certain challenges in using SNMP for RAQMON too. And they are: - The benefit is added flexibility of the proposed by RAQMON Framework could be constrained. - Sending out Acknowledgements from RRCs to RDSs can create bottleneck as additional RDS load is created, specially when the RRCs will be receving many Inform PDUs from many RRcs. - Sending ACKs also wastes network bandwidth. In a reasonable sized Enterprise and Service provider systems this can be a significant amount of load.

To get rid of the Ack as the RDS/RRC protocol which needs not be acknowledgement oriented, SNMP Traps could be used instead of Informs. This will allow one to use SNMP without avoiding performance related issues as mentioned above, with the disadvantage of loss of reliability in passing the information.

6.2 Mapping RAQMON PDUs to RTCP as RDS/RRC Network Transport Protocol

The RAQMON PDU Transfer is comprised of unidirectional exchange of PDUs between RDSs and an RRC. The protocol data units are mapped to a connectionless datagram service (UDP).

As outlined in RFC 1889, an RTCP APP packet allows Applications to defined RTCP packets. Within RTCP framework, a RAQMON PDUs is represented as an Application Specific Report and uses RTCP APP Packets to transport RAQMON PDU.

Figure 4 below shows how RAQMON PDUs can use RTCP APP Packets to transport RAQMON PDUs between RDS and RRC.

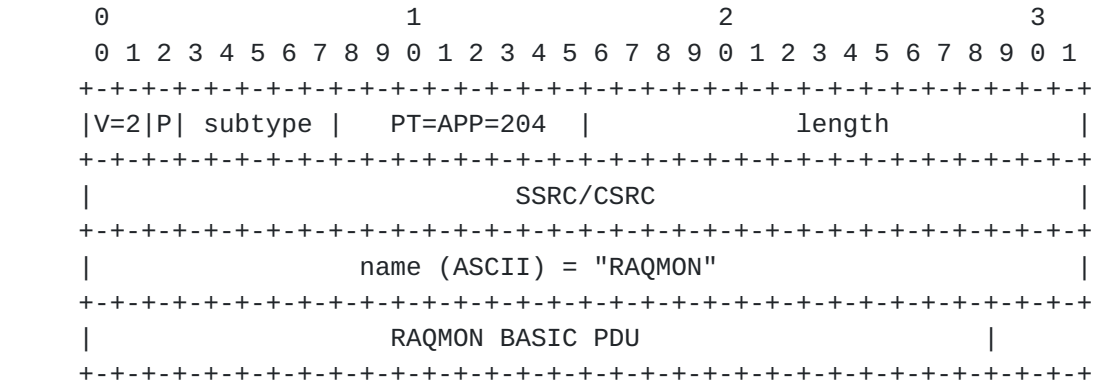


Figure 4 - Using RTCP APP Packets to transport RAQMON PDUs

The RTCP APP packets are intended for experimental use as new applications and new features are developed, without requiring packet type value

registration. To be backward compatible RTCP APP packets used by RAQMON SHOULD be Internet Assigned Numbers Authority (IANA) Registered.

version (V), padding (P), length:
As described for the SR packet (see Section 6.3.1).

subtype: 5 bits
subtype 1 in RAQMON Specific RTCP APP packet SHOULD be used by the BASIC RAQMON PDU and subtype 2 should be preserved for RAQMON APP PDUs. These unique definitions will be IANA registered.

packet type (PT): 8 bits
Contains the constant 204 to identify this as an RTCP APP packet.

name: 4 octets
The name chosen by the RMON WG defining the set of APP packets will be unique with respect to other APP packets and will be IANA Registered as "RAQMON" with all uppercase. The name field in RTCP APP Packet is interpreted as a sequence of four ASCII characters.

application-dependent data: variable length
RAQMON PDUs sent by the RDS in the format specified in Figure 4 will be interpreted by the RAQMON Report Collector (RRC) and not RTP itself. RAQMON PDUs must be a multiple of 32 bits long.

+ During a monitored real-time session, the RDS emits a Report PDU every M seconds toward the RRC as provisioned by the RDS.

+ The RRC collects the Report PDUs and correlate them with its database.

Though this is a simple one-way send protocol, the RDSs will not be capable of inferring whether a PDU was received by the RRC as Report PDUs are transmitted over a lossy network.

So one uses proposed RTCP like protocol as RDS/RRC Network Transport Protocol each Report PDU must contain enough information to uniquely identify the PDUs and correlate to an ongoing session. RRCs could use DSRC field and a unique device ID (i.e. like 6 Octet MAC address or IP Address) to define a unique session.

However this will cause 6-octet overhead worth wasted bandwidth per PDU.

6.2.1 - Pseudo code for RDS & RRC

RDS:

```
           when (session starts} {
             report.identifier = session.endpoints, session.starttime;
             report.timestamp = 0;
             while (session in progress) {
                  wait interval;
                  report.statistics = update statistics;
                  report.curtimestamp += interval;
                  if encryption required
                      report_data = encrypt(report, encrypt parameters);
                  else
                      report_data = report;
                  raqmon_pdu = header, report_data;
                  send raqmon-pdu;
             }
           }
   RRC:
           listen on raqmon port
           when ( raqmon_pdu received ) {
               decrypt raqmon_pdu.data if needed

               if report.identifier in database
                  if report.current_time_stamp > last update
                     update session statistics from report.statistics
                  else
                     discard report
            }
```

6.2.2 Port Assignment

As specified in the previous sections that Transport of RAQMON PDUs can be performed using various underlying network transport protocol like TCP and UDP.

Applications operating under RAQMON Framework may use any unreserved UDP port. For example, a session management program can allocate the port randomly. A single fixed port cannot be required because multiple applications using RAQMON are likely to run on the same host, and there are some operating systems that do not allow multiple processes to use the same UDP port with different multicast addresses.

However, port numbers 5XXX have been registered with IANA for use with those applications that choose to use them as the default port for RAQMON PDUs over RTCP. Hosts that run multiple applications may use this port as an indication to have used RAQMON if they are not subject to the constraint of the previous paragraph.

Applications need not have a default and may require that the port be explicitly specified. The particular port number was chosen to lie in the range above 5000 to accommodate port number allocation practice within the Unix operating system, where privileged processes can only use port numbers below 1024 and port numbers between 1024 and 5000 are automatically assigned by the operating systems.

6.2.3 Reliability

RAQMON framework will allow an RDS to report QOS Parameters to multiple RRCs. Such mechanism would allow better chances of backup and restore QOS parameters. However backup, synchronization of multiple RRCs are beyond the scope of this document is left to the discretion of system designers and implementers.

6.3 Report Aggregation and Statistical Data processing

The RAQMON MIB is designed to provide very simple and minimal aggregations of various RAQMON Parameters defined in table 1. RAQMON MIB is designed to not to provide extensive aggregations like APM MIB [29] or TPM MIB [30] and one should use APM and TPM MIBs to aggregate based on protocols (e.g. Performance of HTTP, RTP) or based on application (e.g. Performance of VoIP, Video Applications).

In RAQMON Framework, RDSs are not burdened by statistical data processing as RDSs may be co-resident in end-devices and could be resource constrained. Various aggregations are performed by the RRC.

Aggregation of RAQMON parameters collected over a period of time is dependent on aggregation algorithms. In the RAQMON MIB, aggregation can be performed only on specific RAQMON metrics parameters specified below:

End-to-End Delay

Inter Arrival Jitter

Cumulative Packet Loss

Packet Loss in Fraction

CPU utilization in Fraction

Memory utilization in Fraction

The aggregation always results in the following statistics:

Mean End-to-End Delay

Min End-to-End Delay

Max End-to-End Delay

Mean Inter Arrival Jitter

Min Inter Arrival Jitter

Max Inter Arrival Jitter

Mean Cumulative Packet Loss

Min Cumulative Packet Loss

Max Cumulative Packet Loss

Mean Packet Loss in Fraction

Min Packet Loss in Fraction

Max Packet Loss in Fraction

Mean CPU utilization in Fraction

Min CPU utilization in Fraction

Max CPU utilization in Fraction

Mean Memory utilization in Fraction

Min Memory utilization in Fraction

Max Memory utilization in Fraction

For this document following aggregation definitions are used:

Mean:

Mean is defined as the statistical average of a metric over the duration of a communication session. For example if End-to-End delay metric of an end device within a communication session is reported N times by the RDS, then the Mean End-to-End Delay is the average End-to-End Delay metric over N entries.

Min:

Min is defined as the statistical minimum of a metric over the duration of a communication session. For example if End-to-End delay metric of an end device within a communication session is reported N times by the RDS, then the Min End-to-End Delay is the minimum of all N End-to-End Delay metric entries in the table.

Max:

Max is defined as the statistical maximum of a metric over the duration of a communication session. For example if End-to-End delay metric of an end device within a communication session is reported N times by the RDS, then the Max End-to-End Delay is the maximum of all N End-to-End Delay metric entries in the table.

6.4 Keeping Historical Data and Storage

It is evident from the document that, RAQMON MIB data need to be managed to optimize storage space. Enormous volume of data gathered in a communication session could be optimized for storage space by performing and storing only aggregated RAQMON metrics for history if required.

Such storage space optimization can be performed in following ways:

1. Store data in the MIB only at the end of a communication session (i.e. after receiving an END packet), the aggregated data could be stored in RAQMON MIB as Mean, Max or Min entry and saved for historical purposes. This will minimize storage space requirement as instead of a column in a table, only few scalars will be used to store a metric.

2. A time based algorithms that aggregate data over a specific period of time within a communication session (i.e. thus requiring less entries) also reduces storage space requirements. For example, if RDS sends data out every 10 seconds and RRC writes to the RAQMON MIB once every minute, for every 6 data points there will be one MIB entry.

3. Clearing up historical data periodically over a calendar time using administration policy can perform further storage space optimization. For example, an administrator could create a policy such that all historical data get cleared up every 60 days. Such policies are interpreted by the application, not by the RRC and usage of these policies left at the application developer's discretion.

7. References

[RFC2571] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, April 1999.

[RFC1155] Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.

[RFC1212] Rose, M., and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.

[RFC1215] M. Rose, "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.

[RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, May 1990.

[RFC1901] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.

[RFC1906] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.

[RFC2572] Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999.

[RFC2574] Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999.

[RFC1905] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network

Management Protocol (SNMPv2)", RFC 1905, January 1996.

[RFC2573] Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", RFC 2573, April 1999.

[RFC2575] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999.

[RFC2570] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, April 1999.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2819] Waldbusser, S., "Remote Network Monitoring Management Information Base", STD 59, RFC 2819, May 2000

[RFC2613] Waterman, R., Lahaye, B., Romascanu, D., and S. Waldbusser, "Remote Network Monitoring MIB Extensions for Switched Networks, Version 1.0", RFC 2613, June 1999

[RFC1213] McCloghrie, K., and M. Rose, Editors, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II", STD 17, RFC 1213, March 1991.

[RFC2863] McCloghrie, K., and Kastenholtz, F., "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC1890] H. Schulzrinne, "RTP Profile for Audio and Video Conferences with Minimal Control" RFC 1890, January 1996.

[RFC1889] Henning Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications" RFC 1889, January 1996.

[RFC1305] Mills, D., "Network Time Protocol Version 3", RFC 1305, March 1992.

[RFC1034] Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987.

[RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

[RFC1597] Rekhter, Y., Moskowitz, R., Karrenberg, D., and G. de Groot, "Address Allocation for Private Internets", RFC 1597, March 1994.

[RFC2679] G. Almes, S.kalidindi and M.Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, September 1999

[RFC2680] G. Almes, S.kalidindi and M.Zekauskas, "A One-way Packet Loss Metric for IPPM", RFC 2680, September 1999

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

[WALDBUSSER] Steven Waldbusser, "Application Performance Measurement MIB", draft-ietf-rmonmib-apm-mib-04.txt, July 2001

[DIETZ] Russel Dietz, Robert Cole, "Transport Performance Metrics MIB", draft-ietf-rmonmib-tpm-mib-03.txt, July 2001

[ISO10646] International Standards Organization, "ISO/IEC DIS 10646-1:1993 information technology -- universal multiple-octet coded character set (UCS) -- part I: Architecture and basic multilingual plane," 1993.

[UNICODE] The Unicode Consortium, The Unicode Standard New York, New York:Addison-Wesley, 1991.

[IEEE802.1D] Information technology-Telecommunications and information exchange between systems--Local and metropolitan area networks-Common Specification a--Media access control (MAC) bridges: 15802-3: 1998 (ISO/IEC) [ANSI/IEEE Std 802.1D, 1998 Edition]

[RFC1349] P. Almquist, "Type of Service in the Internet Protocol Suite", RFC 1349, July 1992

[RFC1812] F. Baker, "Requirements for IP Version 4 Routers" RFC1812, June 1995

[RFC2474] K. Nicholas, S. Blake, F. Baker and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC2474, December 1998

[RFC2475] S. Blake, D. Black, M. Carlson, E.Davies, Z.Wang and W.Weiss, "An Architecture for Differentiated Services" RFC2475, December 1998

[SIDDIQUI1] A. Siddiqui, D.Romascanu, E. Golovinsky, and R. Smith, "Real-time Application Quality of Service Monitoring (RAQMON) MIB", Internet-Draft, draft-siddiqui-rmonmib-raqmon-mib-02.txt, October 2002

[SIDDIQUI2] A. Siddiqui, D.Romascanu, and E. Golovinsky, "Framework for Real-time Application Quality of Service Monitoring (RAQMON)", Internet-Draft, draft-siddiqui-raqmon-framework-00.txt, October 2002

[SIDDIQUI3] A. Siddiqui, D.Romascanu, E. Golovinsky, and R. Smith, "Real-time Application Quality of Service Monitoring (RAQMON) MIB", Internet-Draft, draft-siddiqui-rmonmib-raqmon-mib-01.txt, March 2002

8. Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

9. Security Considerations

There are a number of management objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations.

It is thus important to control even GET access to these objects and possibly to even encrypt the values of these object when sending them over the network via SNMP. Not all versions of SNMP provide features for such a secure environment.

SNMPv1 by itself is not a secure environment. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB.

It is RECOMMENDED that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model [RFC2274] and the View-based Access Control Model [RFC2275] is RECOMMENDED.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

It is also imperative that the RAQMON framework be able to provide the following protection mechanisms:

1. Authentication - the RRC should be able to verify that a RAQMON report was originated by whom ever claims to have sent it.

2. Privacy - RAQMON information include identification of the parties participating in a communication session. RAQMON framework should be able to provide protection from eavsdropping, to prevent an un-authorized third party from gathering potentially sensitive information. This can be achieved by using various payload encryption technologies like DES, 3-DES, AES

3. Protection from Denail of service attacks directed at the RRC - RDSs send RAQMON reports as a side effect of an external event (for example, a phone call is being received). An attacker can try and overwhelm the RRC (or the network) by initiating a large number of events (i.e., calls) for the purpose of swamping the RRC with too many RAQMON PDUs.

To prevent DoS attacks against RRC, the RDS will send the first report for a session only after the session has been in progress for the TBD reporting interval. Sessions shorter than that will not be reported.

4. NAT and Firewall Friendly Design: Presence for IP addresses,

TCP/UDP ports information in RAQMON PDUs may be NAT un-friendly. In such a scenario, where NAT Friendliness is a requirement, the RDS may opt to not to provide IP Addresses in RAQMON PDU. Another way to avoid this problem is by using NAT Aware Application Layer Gateways (ALGs) to fill out IP Addresses in RAQMON PDUs.

10. IANA Consideration

This memo introduces 2 new ports for IANA registration, as specified in Section 6.2.2, at http://www.iana.org/numbers.html

11. Authors' Addresses

Anwar A. Siddiqui
Avaya Labs
307 Middletown Lincroft Road
Lincroft, New Jersey 07738
USA
Tel: +1 732 852-3200
Fax: +1 732 817-5922
E-mail: anwars@avaya.com

Dan Romascanu
Avaya Inc.
Atidim Technology Park, Bldg. #3
Tel Aviv, 61131
Israel
Tel: +972-3-645-8414
Email: dromasca@avaya.com

Eugene Golovinsky
BMC Software
2101 CityWest Blvd.
Houston, Texas 77042
USA
Tel: +1 713 918-1816
Email: eugene_golovinsky@bmc.com

A. Full Copyright Statement