

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 18, 2017

Srivathsa. Sarangapani
Peyush. Gupta
Juniper Networks
V. Hegde
Consultant
Q. Wu
Huawei
July 17, 2016

KPI Metrics for Service Monitoring using TWAMP
draft-spv-ippm-monitor-implementation-services-kpi-02

Abstract

We are using a new method to calculate services KPIs and metrics in the network using TWAMP protocol. This draft outlines the implementation of the service KPIs and there use cases in the service plane in the network. The KPIs discussed in this draft include Service Latency and Application Liveliness detection.

Service latency is defined as the time spent by the packet when it is injected in the service module or service card till the time, serviced packet is received back by the TWAMP server. TWAMP server records the timestamp of the packet when it is injected into the service module and then again record the timestamp when it receives the packet after service is applied in the data plane.

Application Liveliness detection means whether the application is up and running in the network. In case you want to monitor the http application or the dns server and verify if they are up and running, this method is applicable. The implementation can be used for liveliness detection of any service in the network.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 18, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
1.1.	Conventions used in this document	3
1.1.1.	Requirements Language	3
1.2.	Terminology	3
2.	Services KPIs	4
2.1.	Services Keepalive Monitoring	4
2.2.	Service Latency	5
3.	Acknowledgements	7
4.	IANA Considerations	7
5.	Security Considerations	8
6.	Normative References	8
	Authors' Addresses	9

[1.](#) Introduction

The TWAMP-Test runs between a Session-Sender and a Session-Reflector [RFC 5357](#) [[RFC5357](#)]. The existing TWAMP-Test packet format has existing padding octets that are currently not used (either set to zero or pseudo-random values). These octets can be used to carry additional information between the Session-Sender and the Session-Reflector. The proposed extension uses these padding octets and provide a method to monitor services KPIs in the network. This feature is termed as Services KPI Monitoring using TWAMP.

TWAMP server is used to inject the packet in the service plane for calculating the latency and liveliness. This is done as part of TWAMP data connection. The packets being sent out of TWAMP server is a UDP packet carrying the payload for the service for which we are interested in calculating the KPIs. The timestamping is done at the

TWAMP server. Based on the service model, TWAMP server may be running on the same box where the service is hosted or in a remote server.

The Interface between the TWAMP server and the service plane is implementation specific. The underlying transport is UDP since this is in data path. In this draft, the use cases presented are service latency and keep alive monitoring. Service latency for services like DPI, TDF, Video Caching is calculated. Similarly liveliness for http server, dns application is calculated in the implementation part.

As per the proposed extension, both the TWAMP-Control and the TWAMP-Test packet formats are modified. One TWAMP-Test session SHALL be used to monitor KPIs for a specific service. But there can be multiple KPIs monitored using a single test session for a specific service. A single TWAMP-Control connection MAY establish multiple TWAMP-Test sessions that measure KPIs for multiple services in the network.

This extension can be used to monitor KPIs for standalone service or a set of services.

[1.1.](#) Conventions used in this document

[1.1.1.](#) Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#) [[RFC2119](#)].

[1.2.](#) Terminology

TWAMP: Two-Way Active Measurement Protocol

KPI: Key Performance Indicator

DPI: Deep Packet Inspection

CGNAT: Carrier Grade Network Address Translation

SFW: Stateful Firewall

TDF: Traffic Detection Function

DNS: Domain Name Server

HTTP: Hyper Text Transfer Protocol

FTP: File Transfer Protocol

SKMC: Services KPI Monitoring Command

PDU: Protocol Data Unit

[2.](#) Services KPIs

[2.1.](#) Services Keepalive Monitoring

Metric Name: Services Keepalive Monitoring

Metric Description: This indicates whether the service is running or not at any point of time.

Method of Measurement or Calculation: The Session-Reflector SHALL inject the Service PDU to the Service Block for service processing. Based on whether the Session-Reflector received the response, the Session-Reflector SHALL decide whether the Service is alive or not.

Units of Measurement: This metric is expressed as a single bit boolean value. If this bit is set then it indicates that the Service Block is functional. If this bit is NOT set then it indicates that the Service Block is not functional.

Measurement Point(s) with Potential Measurement Domain: This metric is calculated at the Session-Reflector.

Measurement Timing: This metric is an instantaneous value. Based on the kind of service it MAY be a good idea to store the history of this value. It can be stored as an average of last one hour for 24

hours, then average of all values over previous day, week, month, year etc. These data MAY be used for some analytics.

Implementation: The Session-Sender SHALL send the Service PDU as part of the TWAMP-Test Packet Padding. When Session-Reflector receives the TWAMP-Test packet, it SHALL extract the Service PDU. The Session-Reflector SHALL extract the service PDU from the TWAMP Payload and inject it to the service module. For ex, incase of http server, the service PDU can just be a http req. The service module will apply services on this PDU and once service processing is done, it would send the response(http resp) back to Session Reflector. The Session Reflector SHALL now reply back to the TWAMP client/session sender with the actual TWAMP data packet with payload being the boolean flag and response service PDU(http resp).

Verification: The metric value is a boolean which SHOULD be either 0(Service NOT Alive) or 1(Service is Alive).

The Session-Reflector MUST start the Packet Padding with the below 4 octets as indicated below. This is followed by the Service PDU (which MAY be same as whatever was sent by Session-Sender or can be the reply/response packet of the Service Block).

Setting Bit 0(X) indicates that the Session-Reflector successfully sent the Service Request to the Service Block and received the response from the Service Block. If this bit is NOT set then it indicates that the Service Block is not functional.

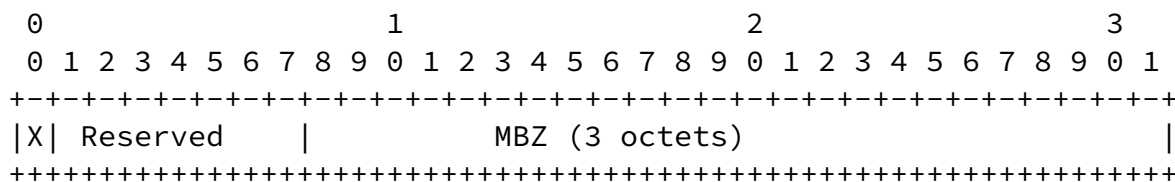


Figure 1: Services Keepalive Monitoring

Use and Applications: This metric is useful for monitoring the liveliness of a service. Normally the liveliness of the Server or a Network Element is not enough to know whether the Service is really Alive or NOT. Say for example if there is a Web Server Application,

then just monitoring whether the Server(where the web server) is alive or not by using ping is just not enough to say whether the application is really Alive or not. There could be instances where in the Server is up and running, while the Web Server Application is not running because of some software bug. These kinds of scenarios can be caught only by probing the application and not just the Server where Applicaton is running.

Reporting Model: This metric needs to be associated with a defined time interval, which could be defined by fixed intervals. Based on need, the TWAMP client can negotiate to check the liveness of a service during connection establishment. If so, then for each of the data packet, the liveness of the service is measured and reported back to session-sender/client for the entire session.

2.2. Service Latency

Metric Name: Service Latency

Metric Description: This indicates the total latency introduced by the service for a data packet which is undergoing specific treatment offered by that service node in the path. Please note that the latency calculation in service agnostic. Service Latency SHALL include the transit time and actual service time. The transit time should refer to round trip time on the path between Session-Reflector

and service node. The service time should refer to service processing time or service treatment time.

Method of Measurement or Calculation: The Session-Reflector SHALL notedown the time: Service latency measurement Sender Timestamp and inject the Service PDU to the Service Block for service processing. When the Session-Reflector receives the response, the Session-Reflector SHALL notedown the time: Service latency measurement Receiver Timestamp.

Units of Measurement: This metric is expressed as a timestamp which is 64 bit value. The format of the timestamp is the same as in [\[RFC1305\]](#) and is as follows: the first 32 bits represent the unsigned integer number of seconds elapsed since 0h on 1 January 1900; the next 32 bits represent the fractional part of a second that has

elapsed since then.

So, Timestamp is represented as follows:

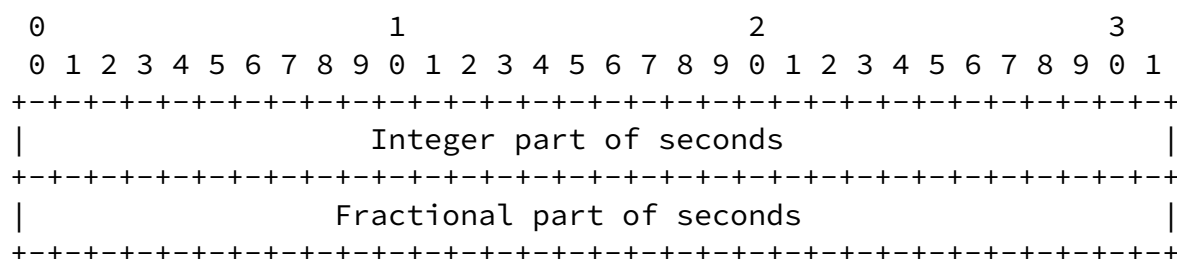


Figure 2: Timestamp Format

Measurement Point(s) with Potential Measurement Domain: This metric is calculated at the Session-Reflector.

Measurement Timing: This metric is an instantaneous value. Based on the kind of service it MAY be a good idea to store the history of this value. It can be stored as an average of last one hour for 24 hours, then average of all values over previous day, week, month, year etc. These data MAY be used for some analytics.

Implementation: The Session-Reflector SHALL extract the service PDU from the TWAMP Payload and inject it to the service module. For ex, incase of http server, the service PDU can just be a http req. The injecting of service module can be broken into below steps:

- Session Reflector should note down the time(say T5) at which this service PDU(http req) is injected to service module.

- The service module will apply services on this PDU and once service processing is done, it would send the response(http resp) back to Session Reflector.
- Once this response serviced PDU is received at Session Reflector, the time(say T6) SHOULD be noted.
- The Session Reflector SHALL now reply back to the TWAMP client/

Value	Description	Explanation
0	None	
1	Keepalive	Whether the respective service is running or not
2	Service Latency	Service Latency which SHALL include the transit time and actual service time

Table 1: TWAMP Services KPIs Registry

Request-TW-Session message defined in [RFC6038].IANA is requested to reserve 2 octets for Service ID as follows:

Value	Description	Semantics	Reference
X	Service ID	2 Octets starting from offset 92th Octet	This document

Table 2: New Services KPIs Monitoring Capability

5. Security Considerations

The TWAMP protocol (RFC 5357) supports authenticated and encrypted mode for TWAMP session and data. The implementation discussed in the proposed extension supports the authenticated and encrypted mode and is therefore provides a secure mechanism to monitor services KPIs in the network.

6. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<http://www.rfc-editor.org/info/rfc5357>>.
- [RFC6038] Morton, A. and L. Ciavattone, "Two-Way Active Measurement Protocol (TWAMP) Reflect Octets and Symmetrical Size Features", RFC 6038, DOI 10.17487/RFC6038, October 2010, <<http://www.rfc-editor.org/info/rfc6038>>.

Authors' Addresses

Srivathsa Sarangapani
Juniper Networks
89, Asthagrama Layout 2nd Stage, Basavehwaranagar
Bangalore 560079
INDIA

Phone: +91 9845052354
Email: srivathsas@juniper.net

Peyush Gupta
Juniper Networks
Flat #206, Keerti Royal Apartment, Outer Ring Road
Bangalore, Karnataka 560043
INDIA

Phone: +91 9449251927
Email: peyushg@juniper.net

Vinayak Hegde
Consultant
Brahma Sun City, Wadgaon-Sheri
Pune, Maharashtra 411014
INDIA

Phone: +91 944984401
Email: vinayakh@gmail.com
URI: <http://www.vinayakhegde.com>

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Phone: +86-25-84565892
Email: bill.wu@huawei.com

