Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 19, 2013

M. Bagnulo
UC3M
T. Burbridge
BT
S. Crawford
SamKnows
P. Eardley
BT
A. Morton
AT&T Labs

January 15, 2013

A registry for commonly used metrics. Independent registries
draft-bagnulo-ippm-new-registry-independent-00

Abstract

This document creates a registry for commonly used metrics, defines the rules for assignments in the new registry and performs initial allocations. This document proposes one particular registry structure with independent registries for each of the fields involved. A companion document draft-bagnulo-ippm-new-registry explores an alternative structure with a single registry with multiple sub-registries.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

This document creates a registry for commonly used metrics. In order to do that, it creates a number of namespaces whose values will be recorded by the registry and will uniquely and precisely identify metrics.

The motivation for having such registry is to allow a controller to request a measurement agent to execute a measurement using a specific metric. Such request can be performed using any control protocol that refers to the value assigned to the specific metric in the registry. Similarly, the measurement agent can report the results of the measurement and by referring to the metric value it can unequivocally identify the metric that the results correspond to.

There was a previous attempt to define a metric registry RFC 4148 [RFC4148]. However, it was obsoleted by RFC 6248 [RFC6248] because it was "found to be insufficiently detailed to uniquely identify IPPM metrics... [there was too much] variability possible when characterizing a metric exactly" which led to the RFC4148 registry having "very few users, if any".

Our approach learns from this, by tightly defining each entry in the registry with only a few parameters open for each. The idea is that the entries in the registry represent different measurement tests, whilst the parameters set things like source and destination addresses that don't change the fundamental nature of the test. The downside of this approach is that it could result in an explosion in the number of entries in the registry. We believe that less is more in this context - it is better to have a reduced set of useful metrics rather than a large set of metrics with questionable usefulness. Therefore this document defines that the registry only includes commonly used metrics that are well defined; hence we require both specification required AND expert review policies for the assignment of values in the registry.

There are a couple of side benefits of having such registry. First the registry could serve as an inventory of useful and used metrics, that are normally supported by different implementations of measurement agents. Second, the results of the metrics would be comparable even if they are performed by different implementations and in different networks, as the metric is properly defined.

This version of the document defines a set of independent registries, that limits the explosion of registry entries by allowing arbitrary combinations of entries in the different entries. The downside is that the list of useful metrics is less defined, as any combination would be defined. Which approach is better is up for discussion.

The registry forms part of a Measurement Plan {do you prefer the term 'Characterization Plan', 'control framework' or 'test schedule'?}. It describes various factors that need to be set by the party controlling the measurements, for example: specific values for the parameters associated with the selected registry entry (for instance, source and destination addresses); and how often the measurement is made. The Measurement Plan might look something like: "Dear measurement agent: Please start test DNS(example.com) and RTT(server.com,150) every day at 2000 GMT. Run the DNS test 5 times and the RTT test 50 times. Do that when the network is idle. Generate both raw results and 99th percentile mean. Send measurement results to collector.com in IPFIX format". The Measurement Plan depends on the requirements of the controlling party. For instance the broadband consumer might want a one-off measurement made immediately to one specific server; a regulator might want the same measurement made once a day until further notice to the 'top 10' servers; whilst an operator might want a varying series of tests (some of which will be beyond those defined in the registry) as determined from time to time by their operational support system. While the registries defined in this document help to define the Measurement Plan its full specification falls outside the scope of this document.

2. The commonly used metrics registry

In this section we define the registry for commonly used metrics. It is composed by the following sub-registries:

- Scheduling registry
- Environment registry
- Output-type registry
- Metric registry

The rationale for the registry structure is to allow flexibility but yet precise definition of metrics. The metric registry defines the metric itself while the other registries define additional aspects that are needed for the measurement plan and that are needed to fully specify a measurement request from a controller to a measurement agent.

2.1. The metrics registry

Each entry for the metrics registry contain the following information:

- Value: A text string that uniquely identifies the metric
- Reference: The specification where the metric is defined

The policy for the assignments in the metric registry is both

specification required AND expert review. This means that in order to create an entry for the metric value a specification defining the metric is required and when that happens, the request for allocation will be reviewed by an expert.

The specification must define the input parameters for the metric as well as the output of the metric. The metric must be well defined, in the sense that two independent implementations must produce uniform and comparable results.

The expert review must make sure that the proposed metric is operationally useful. This means that the metric has proven to be useful in operational/real scenarios.

2.2. The Scheduling registry

Each entry for the scheduling registry contain the following information:

- Value: The name of the scheduling
- Reference: the specification where the scheduling is defined

The scheduling defines the scheduling strategy for the metric. Simplest is Singleton scheduling, where an atomic measurement is made. Other strategies make a series of atomic measurements in a "sample" or "stream", with the schedule defining the timing between each distinct measurement. Each atomic measurement could consist of sending a single packet (such as a DNS request) or sending several packets (for example a webpage). A scheduling strategy requires input parameter(s). Assignment in this registry follows the specification required policy.

2.3. The Environment registry

Each entry for the environment registry contain the following information:

- Value: The name of the environment
- Reference: the specification where the environment is defined

The environment defines the conditions where the metric is expected to be used. It does not define the metric itself, but the context where the metric is executed. Assignment in this registry follows the specification required policy.

2.4. The Output type registry

Each entry for the output type registry contain the following information:

- o Value: The name of the output type
- o Reference: the specification where the output type is defined

The output type define the type of output that the metric produces. It can be the raw results or it can be some form of statistic. Assignment in this registry follows the specification required policy. The specification of the output type must define the format of the output.

3. Initial assignment for the Scheduling registry

3.1. Common parameter definitions

Although each IPPM RFC defines individual parameters and uses them consistently, the parameter names are not completely consistent across the RFC set. For example, the variable "dT" is used in several different ways. This memo uses one set of parameter names, and the reader is cautioned to map the names according to their definitions.

We define some parameters that are used by several types of scheduling:

- o T0: time to begin a test
- o Tf: time to end a test

T0 and Tf are both in seconds and use the date (yyyy-mm-dd) and NTP 64 bit timestamp. T0 includes any control handshaking before the test stream or singleton. Tf is the time the last test data is sent.

As a result, we have:

- o Time when test devices may close the test socket: Tf + Waiting Time (the time to wait before declaring a packet lost is fixed for each metric)
- o Total duration of the test: Tf - T0 + Waiting Time

3.2. Poisson scheduling

The values for this entry are as follows:

- o Value: Poisson
- o Reference: draft-bagnulo-ippm-new-registry

The Poisson scheduling is defined in section 11.1.1 of RFC 2330 [RFC2330] and needs input parameters:

- T0 and Tf: defined above
- lambda: the parameter defining the Poisson distribution. Lambda is the mean number of distinct measurements per second in the sample.

3.3. Periodic scheduling

The values for this entry are as follows:

- Value: Periodic
- Reference: draft-bagnulo-ippm-new-registry

The Periodic sampling is defined in RFC 3432 [RFC3432]. The additional input parameters for the metric required by Periodic scheduling are:

- T0 and Tf: defined above
  * Note that with Periodic sampling, T0 MUST NOT be strictly periodic with other tests of the same type. RFC 3432 [RFC3432] requires randomized start times and describes one way to accomplish this. Also, the duration of the test MUST be limited.
- incT: the time in seconds between one distinct event and the next, where events typically result in repeating singleton measurements of various types (illustrated below).
  * for a periodic stream this is the time between packets in the sample, first bit to first bit
  * for measurements on a process this is the time between the first packets of the process, for example first bit to first bit of the SYN in a TCP 3-way handshake

3.4. Singleton scheduling

The values for this entry are as follows:

- Value: singleton
- Reference: draft-bagnulo-ippm-new-registry

The singleton scheduling covers the case when an atomic metric is performed as per RFC 2330 [RFC2330]. The additional input parameter for the metric required by Singleton scheduling is:

- T0: defined above

4. Initial assignments for the Output Type registry

4.1. Raw

The values for this entry are as follows:

- Value: Raw
- Reference: draft-bagnulo-ippm-new-registry

The results of the metric are delivered in the exact way they are produced by the measurements without any further processing or filtering.

4.2. Xth percentile interval

The values for this entry are as follows:

- Value: Xth-percentile
- Reference: draft-bagnulo-ippm-new-registry

The additional input parameter for the metric is:

- X: the percentile (e.g, if the X input parameter is 99, then the output will be the 99th percentile interval.)

The output when using this Output type will be a a couple of values, expressed in the same units as the raw output, that is the Xth percentile interval, as defined in section 1.3 of RFC 2330 [RFC2330].

4.3. Xth percentile mean

The values for this entry are as follows:

- Value: Xth-percentile-mean
- Reference: draft-bagnulo-ippm-new-registry

The additional input parameter for the metric is

- X: the percentile (e.g, if the X input parameter is 99, then the output will be the 99th percentile mean.)

The output when using this Output type will be a single value, expressed in the same units as the raw output, that is the mean of the sample only considering the values contained in the Xth percentile interval, as defined in RFC 2330 [RFC2330].

5. Initial assignments for the Environment registry

5.1. Undefined

The values for this entry are as follows:

- Value: Undefined
- Reference: draft-bagnulo-ippm-new-registry

The undefined environment is the case where no additional environment settings are defined to perform the metric.

5.2. No cross traffic

The values for this entry are as follows:

- Value: No-cross-traffic
- Reference: draft-bagnulo-ippm-new-registry

It is often important that there is no other traffic than the one generated by the measurement itself while doing the measurement. The reasons for this are two-folded, first, it is sometimes important that the traffic created by the measurement doesn't impact the experience of the users of the measured resource. Second it is sometimes important that no other traffic interferes with the measurement. This can be ensured by checking that the level of user traffic is either zero or low enough to be confident that it won't impact or be impacted by the measurement.

The "No cross traffic" condition is satisfied when, during the 5 seconds preceding measurement of the metric:

- the level of traffic flowing through the interface that will be used to send measurement packets in either direction is less than a threshold value of 1% of the line rate of the aforementioned interface.

The "cross traffic" measurement is made at the interface, associated with the measurement agent, that user traffic flows across. For example, if the probe is attached to the home gateway, then the interface is the service demarcation point where the subscriber connects their private equipment or network to the subscribed service.

Note that the No-cross traffic condition is defined only for the link directly attached to the measurement agent initiating the measurement. There is nothing mentioned about cross traffic on other parts of the path used by measurement packets. In the case the bottleneck of the path is other link than the one directly attached to the device running the measurement agent, it may affect and be affected by the measurement even if the No cross traffic as defined here holds.

DISCUSSION

- It is not clear we need a registry for this. If the only thing we are going to define is the No cross traffic condition, we can simply set it as an input parameter in each metric.
- clarify whether traffic for each direction is less than threshold, or the sum

- current SamKnows probes measure cross-traffic before the measurement of the metric. Another approach would be to measure cross-traffic during the time the metric is measured. Or a hybrid approach. These would either be separate environment entries, or parameterise the existing one.
- current SamKnows probes define a fixed threshold. it could be a parameter
- could ignore broadcast traffic (think SamKnows includes)
- It would be possible to define this a bit more precisely as follows:
  - The "No cross-traffic" condition is defined for active measurements. The measurement agent runs in a device that has one or more interfaces. In active measurements, the measurement agent sends one or more packets. Lets call if0 the interface through with the packets resulting from the measurement are sent through. The no cross traffic condition is fulfilled when during the 5 seconds prior sending each of the packets of the measurement:
    - The traffic incoming through if0 that does not belong to the measurement is lower than 1% of the line rate of if0
    - The traffic coming through the rest of the interfaces towards if0 is less than 1% of the line rate of if0.

6. Initial assignments for the Metric registry

6.1. Comment

Need to work through that we only define T0 and Tf (and not T, dT).

6.2. UDP related metrics

RFC 2681 [RFC2681] defines a Round-trip delay metric and RFC 6673 [RFC6673] defines a Round-trip packet loss metric. We build on these two metrics by specifying several of the open parameters to precisely define several metrics for measuring UDP latency and packet loss. All the UDP related metrics defined in this section use the following:

P-Type:

- IPv4 header values:
  - DSCP: set to 0
  - TTL set to 255
  - Protocol: Set to 17 (UDP)
- UDP header values:
  - Checksum: the checksum must be calculated

- Payload
  - Sequence number: 8-byte integer
  - Timestamp: 8 byte integer. Expressed as 64-bit NTP timestamp as per section 6 of RFC 5905 [RFC5905]
  - No padding

Timeout: 3 seconds

6.2.1. UDP latency metric

We define the UDP latency metric as follows:
- Value: UDP_Latency
- Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined as Type-P-Round-trip-Delay- in RFC 2681 [RFC2681] using the P-Type and Timeout defined above.

The input parameters for this metric are:
- Source IP Address
- Destination IP Address
- Source UDP port
- Destination UDP port
- Time

The output of this metric is the couple of values formed by the timestamp of the sent packet and the time when the echo was received. They are expressedin seconds and use the date (yyyy-mm-dd) and NTP 64 bit timestamp

6.2.2. UDP packet-loss metric

We define the UDP packet-loss metric as follows:
- Value: UDP_packet_loss
- Reference: draft-bagnulo-ippm-new-registry

This metric is defined as Type-P-Round-trip-Loss in RFC 6673 [RFC6673] using the P-Type and Timeout defined above.

The input parameters for this metric are:
- Source IP Address
- Destination IP Address
- Source UDP port
- Destination UDP port
- Time T

The output of this metric is a single value 0 (packet was lost) or 1 (packet has arrived before timeout)

7. ICMP related metrics

RFC 6673 [RFC6673] defines a Round-trip packet loss metric. We build on that metrics by specifying several of the open parameters to precisely define a metric for measuring ICMP packet loss. The ICMP related metric defined in this document use the following:

P-Type:

- o IPv4 header values:
  - * DSCP: set to 0
  - * TTL set to 255
  - * Protocol: Set to 1 (ICMP)
- o ICMP header values:
  - * Type: 8 (Echo request)
  - * Code: 0

Observation: reply packets will contain an ICMP type of 0 Echo reply.

Timeout: 3 seconds

7.1. ICMP packet-loss metric

We define the ICMP packet-loss metric as follows:

- o Value: ICMP_Packet_Loss
- o Reference: draft-bagnulo-ippm-new-registry

This metric is defined as Type-P-Round-trip-Loss in RFC 6673 [RFC6673] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Time T

The output of this metric is a single value 0 (packet was lost) or 1 (packet has arrived before timeout)

8. DNS related metrics

RFC 2681 [RFC2681] defines a Round-trip delay metric. We build on that metric by specifying several of the open parameters to precisely define a metric for measuring DNS latency. The metric uses the following parameters:

P-Type:

- o IPv4 header values:
  - * DSCP: set to 0
  - * TTL set to 255
  - * Protocol: Set to 17 (UDP)
- o UDP header values:
  - * Source port: 53
  - * Destination port: 53
  - * Checksum: the checksum must be calculated
- o Payload: The payload contains a DNS message as defined in RFC 1035 [RFC1035] with the following values:
  - * The DNS header section contains:
    - + QR: set to 0 (Query)
    - + OPCODE: set to 0 (standard query)
    - + AA: not set
    - + TC: not set
    - + RD: set to one (recursion desired)
    - + RA: not set
    - + RCODE: not set
    - + QDCOUNT: set to one (only one entry)
    - + ANCOUNT: not set
    - + NSCOUNT: not set
    - + ARCOUNT: not set
  - * The Question section contains:
    - + QNAME: the FQDN provided as input for the test
    - + QTYPE: the query type provided as input for the test
    - + QCLASS: set to IN
  - * The other sections do not contain any Resource Records.

Observation: reply packets will contain a DNS response and may contain RRs.

Timeout: 3 seconds

8.1. DNS latency metric

We define the DNS latency metric as follows:

- o Value: DNS_Latency
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined as Type-P-Round-trip-Delay in RFC 2681 [RFC2681] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address (the address of the DNS server to be tested)

- QTYPE: A RR
- FQDN: a valid FQDN that will be queried for.
- Time T

The output of this metric is the timestamp when the packet was sent and the delay that it took to receive a response. Please note that any DNS response is valid, including no records in the answer. (Should we be more explicit about what is the output when there is no reply packet received?)

9. Some examples of measurement plans

A measurement plan will be characterized by the following tuple: (Metric, environment, scheduling, output format). We will next present some measurement plans that are currently used.

A measurement plan for measuring the 99th percentile interval of the UDP latency without cross traffics, using a Poisson stream with rate l pkts/sec, stating at time T0 and ending at Tf seconds, between source IP address IPs and source port Ps and destination IP address IPd and destination port Pd would be expressed as:

```
(UDP_Latency(IPs,Ps,IPd,Pd), No-cross-traffic, Poisson(T0,Tf,l),
Xth-percentile(99))
```

A measurement plan for measuring the UDP packet loss ration without cross traffics, using a Poisson stream with rate l pkts/sec, stating at time T0 and ending at Tf seconds, between source IP address IPs and source port Ps and destination IP address IPd and destination port Pd would be expressed as:

```
(UDP_Packet_Loss(IPs,Ps,IPd,Pd), No-cross-traffic,
Poisson(T0,Tf,l), Xth-percentile-mean(100))
```

A measurement plan for measuring the ICMP packet loss ratio, using a Periodic stream s second between packets, stating at time T0 and ending at Tf seconds, between source IP address IPs and destination IP address IPd would be expressed as:

```
(ICMP_Packet_Loss(IPs,IPd), Undefined, Periodic(T0,Tf,s), Xth-
percentile-mean(100))
```

A measurement plan for measuring the DNS latency for resolving FQDN foo.com between a resolver in IP address IPs and a server with address IPd at time T would be expressed as:

```
(DNS_Latency(IPs,IPd,foo.com), Undefined, Singleton(T), raw)
```

10. Security considerations

TBD

11. IANA Considerations

TBD

12. Acknowledgments

We would like to thank Henning Schulzrinne for many constructive comments and input on early versions of this document.

13. References

13.1. Normative References

[RFC2330]   Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.

[RFC3432]   Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, November 2002.

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

[RFC6673]   Morton, A., "Round-Trip Packet Loss Metrics", RFC 6673, August 2012.

[RFC1035]   Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC5905]   Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

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

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

[RFC3393]   Demichelis, C. and P. Chimento, "IP Packet Delay Variation

Metric for IP Performance Metrics (IPPM)", RFC 3393, November 2002.

[RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, March 2009.

13.2. Informative References

[RFC4148] Stephan, E., "IP Performance Metrics (IPPM) Metrics Registry", BCP 108, RFC 4148, August 2005.

[RFC6248] Morton, A., "RFC 4148 and the IP Performance Metrics (IPPM) Registry of Metrics Are Obsolete", RFC 6248, April 2011.

Authors' Addresses

Marcelo Bagnulo
Universidad Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
SPAIN

Phone: 34 91 6249500
Email: marcelo@it.uc3m.es
URI: http://www.it.uc3m.es

Trevor Burbridge
British Telecom
Adastral Park, Martlesham Heath
Ipswich
ENGLAND

Email: trevor.burbridge@bt.com

Sam Crawford
SamKnows

Email: sam@samknows.com

Philip Eardley
British Telecom
Adastral Park, Martlesham Heath
Ipswich
ENGLAND

Email: philip.eardley@bt.com

Al Morton
AT&T Labs
200 Laurel Avenue South
Middletown, NJ
USA

Email: acmorton@att.com