Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 25, 2013

M. Bagnulo
UC3M
B. Trammell
ETH Zurich

February 21, 2013

An LMAP application for IPFIX
draft-bagnulo-lmap-ipfix-01

Abstract

This document explores the possibility of using IPFIX to report test results from a Measurement Agent to a Collector, in the context of a large measurement platform.

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 August 25, 2013.

Copyright Notice

Table of Contents

1. Introduction

A Large-scale Measurement Platform (LMP) is composed by the following fundamental elements: a set of Measurement Agents (MAs), one or more Controllers and one or more Collectors. There may be additional elements in any given such of these platforms, but these three elements are present in all of them. The MAs are pieces of code that run in specialized hardware (hardware probes) or in general purpose devices such as PCs, laptops or mobile phones (software probes). The MA run the tests against other MAs distributed across the Internet. Typically most of the MAs are located in end user networks and a few MAs are located deep into the ISP network, and typically tests are executed from the MAs in the periphery towards MAs located in the core. The Controller is the element that controls the MAs and informs the MAs about what tests to do and when to do them. The protocol between the Controller and the MA is called the Control protocol. After performing the tests, the MAs send the data about the results of the tests performed to the Collector. The protocol used to report test result data from the MA to the Collector is called the Report protocol. In this document we explore the possibility of using IPFIX [I-D.ietf-ipfix-protocol-rfc5101bis] as a Report protocol for large scale measurement platforms.

1.1. A quick introduction to IPFIX

IPFIX [I-D.ietf-ipfix-protocol-rfc5101bis] is a unidirectional, transport-independent export protocol for binary data records, with a focus on network measurement and operations applications. The structure of the data records is described in-band by Templates, which refer to Information Elements (IEs) from a common information model managed by IANA [ipfix-iana]. The basic IEs cover most Layer 3 and Layer 4 measurement needs, and the information model can be extended [I-D.ietf-ipfix-ie-doctors] as well as supplemented by private IEs.

IPFIX organizes data records into Messages. A Message is a sequence of Sets preceded by a Message Header which, among other things, includes an Observation Domain ID (roughly, identifying where the records in the Message were measured) and an Export Time (when the Message was originally sent).

A Set contains Records preceded by a Set Header, which contains a Set ID identifying the type of the records the Set contains. Template Sets, idenfied by a special Set ID, contain Templates, which are sequences of IE identifiers and lengths; these define the fields of the records they describe. A Template’s ID matches the Set ID of the Sets containing records described by the Template.

On-wire data structures in IPFIX are fully discussed in section 3 of [I-D.ietf-ipfix-protocol-rfc5101bis].

Since many records may be described by a single Template, IPFIX's data representation is more efficient than those based on inline record structures (e.g. XML, JSON). Additionally, this arrangement implies that a device that only needs to export one or two fixed-length record types can implement IPFIX with minimal code supporting fixed message and set lengths with fixed-length templates.

IPFIX also supports a feature called Options Templates. An Options Template allows a data record to be scoped to a set of values of particular IEs (called its Scope). For example, a set of test parameters could be scoped to a test identifier IE, and that test identifier exported in a record together with the results. This mechanism allows more efficient data export, as explored in Section 4 below; more information is available in [RFC5473].

1.2. Applying IPFIX to LMAP

In IPFIX terminology [RFC5470], the MA encompasses both the Metering Process (MP) and the Exporting Process (EP), while the Collector is the Collecting Process (CP). IPFIX is used between the EP/MA and the Collector/CP. We propose LMA as an application of IPFIX per [I-D.ietf-ipfix-ie-doctors].

Some considerations about the use of IPFIX for LMP:

- Separation between Control and Report Protocols: Within a single measurement platform, different protocols can be used for Control and Report, though they must share a common vocabulary representing the measurements to be performed. In particular, if a platform implements IPFIX as a Report protocol, it must implement a different protocol (e.g. NETCONF or other) as a Control protocol.
- Report protocol diversity: Some platforms may use IPFIX as a Report protocol, while other platforms may decide to use other protocols (e.g. the Broadband forum architecture may decide to use a different one). We believe that it is important to support this protocol diversity. A key element to support such diversity is an independent metric registry (see [I-D.bagnulo-ippm-new-registry-independent]) where values for metric identifiers are recorded independently of the Control and/or Report protocol is used. This affects how we use IPFIX as a Report protocol, as presented in this document.
- Minimal IPFIX implementation: The unidirectional nature of the protocol and simple wire format make minimal implementations of Exporting Processes possible. These minimal implementations are well suited to small-scale MAs (such as a mobile app or a process

running in a home router). These only need to know about the specific Templates supporting the metric(s) to be reported.

2. Using IPFIX to report test results

In order to use IPFIX to report test results from the MA to the Collector, we need first to understand what information needs to be conveyed. The information transmitted by the MA to the Collector when reporting test(s) results is the following:

- Information about the MA: in particular a MA identifier
- Information about the time of the report: when the report was sent (not necessarily when the test was performed)
- Information describing the test. This includes:
  - An identifier of the metric used for the test (see the Metric registry of [I-D.bagnulo-ippm-new-registry-independent])
  - An identifier of the scheduling strategy used to perform the test (see the Scheduling registry of [I-D.bagnulo-ippm-new-registry-independent]) and potential input parameters for the schedule, such as the rate.
  - An identifier of the output format, (see the Output Type registry of [I-D.bagnulo-ippm-new-registry-independent])
  - An identifier of the environment, notably, if cross traffic was or not present during the execution of the test. (see the Environment registry of [I-D.bagnulo-ippm-new-registry-independent])
  - The input parameters for the test, such as source IP address, destination IP address, source and destination ports and so on.
- Information describing the test results. This widely varies with each test, but can include time each packet was sent and received, number of sent and lost packets or other information.

We next explore how we can encode this information in IPFIX.

In order to convey test information using IPFIX we will naturally use the IPFIX message format and we will define a Template describing the records containing the test result data. We will re-use as many already defined Information Elements (IEs) as possible and we will identify new IEs that are needed.

Part of the information can be conveyed using the fields in the IPFIX header, namely:

- Information about the MA: In order to convey the MA identifier we can use the Observation Domain field present in the IPFIX header. This would allow to have up to 2^32 MA, which seems sufficient.
- Information about the time of the report: The IPFIX header contains an Export Time field that can be used to convey this information.

The information describing the test is included in a Template set that contains multiple IEs for each of the different pieces of information we need to convey. This includes:

- An identifier of the metric used for the test. In order to convey that we need to define a new IE, let's call it metricIdentifier. The values for this element will be the values registered in the Metric registry of [I-D.bagnulo-ippm-new-registry-independent].
- An identifier of the scheduling strategy used to perform the test. Again, this will be a new IE, called testSchedule and its values will be the values defined in the Scheduling registry of [I-D.bagnulo-ippm-new-registry-independent]. The potential input parameters for the schedule, such as the rate, we probably need a new IE for each of these. Usual scheduling distributions only require a rate, so we can define a new IE called scheduleRate which value will contain the rate for the requested distribution.
  * NOTE: The distribution in some cases could be extracted from the results, for example, if the results contain each packet sent, it would be easy to spot a periodic scheduling. Probably not so obvious for the Poisson one. Maybe this would be an optional element to be carried when it is not possible to extract it from the test results.
- An identifier of the output format. A new IE outputType is needed for this and it would take values out of the ones in the Output Type registry of [I-D.bagnulo-ippm-new-registry-independent]. Some of the output formats require an additional input, like the percentile used to trim the outliers when performing means. There are two approaches here. One approach is that the the Output Type registry creates different entries for the different percentiles, which would result in more entries in the Output Type registry (e.g. one entry for the 95th percentile mean and another one for the 90th percentile mean). This may cause an increase number of entries in the Output Type registry, but since there are not too many usual values, it is likely to be manageable. The other approach is to define an additional IE, for instance, the percentile IE that will have the values for the different percentiles used in the output.
- An identifier of the environment, notably, if cross traffic was or not present during the execution of the test. Again, a new IE is needed for this testEnvironment. It will take values of the the Environment registry of [I-D.bagnulo-ippm-new-registry-independent].
- The input parameters for the test. Most of these can be expressed using existing IEs, such as sourceIPv4Address, destinationIPv4Address, etc.

Information describing the test results. This widely varies with each test, but can include time each packet was sent and received, number of sent and lost packets or other information. Again most of

these can be expressed using existent IEs, and some new ones can be defined if needed for a particular test.

3. Example: UDP latency test

Let's consider the example of UDP latency. Suppose a MA wants to report the results of a UDP latency test, performed from its own IP address (e.g. 192.0.2.1) to a destination IP address (e.g. 203.0.113.1), using source port 23677 and destination port 34567. The test is performed using a periodic scheduling with a rate of 1 packet per second during 3 seconds and starts at 10:00 CEST. The test was performed without cross-traffic and the output type is raw.

The Template for this would be:

- metricIdentifier
- testSchedule
- scheduleRate
- outputType
- testEnvironment
- sourceIPv4Address
- destinationIPv4Address
- sourceTransportPort
- destinationTransportPort
- flowStartMilliseconds
- flowEndMilliseconds

The data set following this template for the example would be:

- metricIdentifier = UDP_Latency as per [I-D.bagnulo-ippm-new-registry-independent]
- testSchedule = Periodic as per [I-D.bagnulo-ippm-new-registry-independent]
- scheduleRate = 1
- outputType = Raw as per [I-D.bagnulo-ippm-new-registry-independent]
- testEnvironment = No-cross-traffic as per [I-D.bagnulo-ippm-new-registry-independent]
- sourceIPv4Address = 192.0.2.1
- destinationIPv4Address = 203.0.113.1
- sourceTransportPort = 23677
- destinationTransportPort = 34567
- flowStartMilliseconds = 08:00:00.000 UTC
- flowEndMilliseconds = 08:00:00.001 UTC

- metricIdentifier = UDP_Latency as per [I-D.bagnulo-ippm-new-registry-independent]

```
      testSchedule = Periodic as per
      [I-D.bagnulo-ippm-new-registry-independent]
      scheduleRate = 1
      outputType = Raw as per
      [I-D.bagnulo-ippm-new-registry-independent]
      testEnvironment = No-cross-traffic as per
      [I-D.bagnulo-ippm-new-registry-independent]
      sourceIPv4Address = 192.0.2.1
      destinationIPv4Address = 203.0.113.1
      sourceTransportPort = 23677
      destinationTransportPort = 34567
      flowStartMilliseconds = 08:00:01.000 UTC
      flowEndMilliseconds = 08:00:01.002 UTC
      --------------------------
      metricIdentifier = UDP_Latency as per
      [I-D.bagnulo-ippm-new-registry-independent]
      testSchedule = Periodic as per
      [I-D.bagnulo-ippm-new-registry-independent]
      scheduleRate = 1
      outputType = Raw as per
      [I-D.bagnulo-ippm-new-registry-independent]
      testEnvironment = No-cross-traffic as per
      [I-D.bagnulo-ippm-new-registry-independent]
      sourceIPv4Address = 192.0.2.1
      destinationIPv4Address = 203.0.113.1
      sourceTransportPort = 23677
      destinationTransportPort = 34567
      flowStartMilliseconds = 08:00:02.000 UTC
      flowEndMilliseconds = 08:00:02.001 UTC
      --------------------------
```

4. Example: UDP latency test with Options

In the previous example, the test description is exported together with the results in the record. If a particular set of test parameters will be repeated often by a given MA, the common properties can be grouped into an Options record, described by an Options Template and identified by a new Information Element, with Data Records referring back to this identifier.

In this case, two templates are used: an Options Template to

The Options Template would be:

```
   testParametersId {scope}
   metricIdentifier
```

```
      testSchedule
      scheduleRate
      outputType
      testEnvironment
      sourceIPv4Address
      destinationIPv4Address
      sourceTransportPort
      destinationTransportPort
```

The Template for each Data Record carrying results would be:

```
      testParametersId {scope}
      flowStartMilliseconds
      flowEndMilliseconds
```

The data set carrying the common properties would be:

```
      testParametersId = 1
      metricIdentifier = UDP_Latency as per
      [I-D.bagnulo-ippm-new-registry-independent]
      testSchedule = Periodic as per
      [I-D.bagnulo-ippm-new-registry-independent]
      scheduleRate = 1
      outputType = Raw as per
      [I-D.bagnulo-ippm-new-registry-independent]
      testEnvironment = No-cross-traffic as per
      [I-D.bagnulo-ippm-new-registry-independent]
      sourceIPv4Address = 192.0.2.1
      destinationIPv4Address = 203.0.113.1
      sourceTransportPort = 23677
      destinationTransportPort = 34567
      ---------------------------
```

And the data set carrying results would be:

```
      testParametersId = 1
      flowStartMilliseconds = 08:00:00.000 UTC
      flowEndMilliseconds = 08:00:00.001 UTC
      ---------------------------
      testParametersId = 1
      flowStartMilliseconds = 08:00:01.000 UTC
      flowEndMilliseconds = 08:00:01.002 UTC
      ---------------------------
      testParametersId = 1
      flowStartMilliseconds = 08:00:02.000 UTC
      flowEndMilliseconds = 08:00:02.001 UTC
      ---------------------------
```

This approach sacrifices some complexity at the MA (which must assign testParametersIds and use multiple Templates) and the collector (which must track testParametersId of each set of parameters to

reassemble "complete" results) to gain export efficiency. A quantitative measurement of efficiency gains and tradeoffs for a set of specified result records will follow in a future version of this draft.

5. What standardization is needed for this?

So, in order to enable the use of IPFIX for LMP, the following pieces of standardization would be required.

- The definition of the metric registry. This is not specific for IPFIX as any other Report protocol is likely to require this, but having an independent registry enables multiple report protocols.
- The definition of new IEs. Some of them are identified above, some other are likely to be needed as well.
- The definition of the Templates sets for each of the tests to be performed. This is necessary to have a defined Template that different vendors can implement and can use the IPFIX format in the wire, but they don’t need to fully implement IPFIX parsing to read arbitrary Template sets, just the ones associated with the relevant metrics.

6. Security considerations

The security requirements for the protocol between the MA and the collector have been identified in [I-D.eardley-lmap-framework] and in [I-D.schulzrinne-lmap-requirements]. The identified requirements are:

- Mutual authentication and authorization between the MA and the collector. This means that the collector must be able to verify the identity of the MA and to also verify that the MA is authorized to feed data into the collector and that the MA must be able to verify the identity of the collector and recognize it as a valid collector for the data it is reporting.
- The information flowing between the MA and the collector must be confidential.
- The integrity of the information flowing from the MA and the collector must be protected.

Not surprisingly these are exactly the same requirements imposed to the design of the IPFIX protocol, in particular for the flow of data between the EP and the CP. As described in the security considerations of IPFIX [I-D.ietf-ipfix-protocol-rfc5101bis], IPFIX address these requirements by imposing the use of TLS or DTLS with mutual authentication though certificates. The authorization relies on having a list of authorized MAs in the collector and a list of collectors in the MAs, identified by information in the Distinguished

Name and/or Common Name of their certificate. Current IPFIX specifications and implementations already support TLS and DTLS and this covers the aforementioned requirements. We are aware that some of the current platforms use ssh as a transport protocol between the MAs and the collector. Using ssh allow avoiding the use of certificates, but may result in a more complex key management (which may not be an issue in certain deployments). We believe it would be possible to define an ssh transport for IPFIX if deemed necessary.

IPFIX recommends the use DNS-IDs in the certificates, which applies to EPs and CPs with relatively static addressing. This is probably not a good fit for MAs, since they are likely to have a dynamic address. In this draft we have proposed to use the Observation domain as identifier for the MAs. While the Observation domain must not be globally unique within IPFIX, it would be possible to make it so in a particular measurement platform. The Observation Domain Identifier could then appear in the Common Name of the certificate in some form. Additionally, access control in very large deployments could rely not on identifying specific MAs, but on ensuring that a peer MA or collector had a certificate signed by one of a set of specified authorized issuers.

7. IANA Considerations

TBD

8. Acknowledgements

We would like to thank Sam Crawford and Al Morton for input on early discussions for this draft.

9. References

9.1. Normative References

[I-D.ietf-ipfix-protocol-rfc5101bis]
Claise, B. and B. Trammell, "Specification of the IP Flow Information eXport (IPFIX) Protocol for the Exchange of Flow Information", draft-ietf-ipfix-protocol-rfc5101bis-06 (work in progress), February 2013.

[RFC5470] Sadasivan, G., Brownlee, N., Claise, B., and J. Quittek, "Architecture for IP Flow Information Export", RFC 5470, March 2009.

[I-D.bagnulo-ippm-new-registry-independent]
Bagnulo, M., Burbridge, T., Crawford, S., Eardley, P., and A. Morton, "A registry for commonly used metrics. Independent registries", draft-bagnulo-ippm-new-registry-independent-00 (work in progress), January 2013.

[ipfix-iana]
Internet Assigned Numbers Authority, "IP Flow Information Export (IPFIX) Entities", IANA IPFIX Registry , February 2013.

9.2. Informative References

[RFC5473] Boschi, E., Mark, L., and B. Claise, "Reducing Redundancy in IP Flow Information Export (IPFIX) and Packet Sampling (PSAMP) Reports", RFC 5473, March 2009.

[I-D.ietf-ipfix-ie-doctors]
Trammell, B. and B. Claise, "Guidelines for Authors and Reviewers of IPFIX Information Elements", draft-ietf-ipfix-ie-doctors-07 (work in progress), October 2012.

[I-D.eardley-lmap-framework]
Eardley, P., Burbridge, T., and A. Morton, "A framework for large-scale measurements", draft-eardley-lmap-framework-00 (work in progress), February 2013.

[I-D.schulzrinne-lmap-requirements]
Schulzrinne, H., Johnston, W., and J. Miller, "Large-Scale Measurement of Broadband Performance: Use Cases, Architecture and Protocol Requirements", draft-schulzrinne-lmap-requirements-00 (work in progress), September 2012.

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

Brian Trammell
Swiss Federal Institute of Technology Zurich
Gloriastrasse 35
8092 Zurich
Switzerland

Email: trammell@tik.ee.ethz.ch