IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 28, 2008

S. Niccolini
S. Tartarelli
J. Quittek
NEC
M. Swany
UDel

February 25, 2008

Information Model and XML Data Model for Traceroute Measurements
draft-ietf-ippm-storetraceroutes-08

Status of this Memo

Copyright Notice

Abstract

This document describes a standard way to store the configuration and the results of traceroute measurements. This document first of all describes the tool itself; afterwards, the common information model is defined dividing the information elements in two semantically separated groups (configuration elements and results ones). Moreover

an additional element is defined to relate configuration elements and results ones by means of a common unique identifier. On the basis of the information model a data model based on XML is defined to store the results of traceroute measurements.

Table of Contents

1. Introduction

Traceroutes are being used by lots of measurement efforts, either as an independent measurement or to get path information to support other measurement efforts. That is why there is the need to standardize the way the configuration and the results of traceroute measurements are stored. The standard metrics defined by IPPM working group in matter of delay, connectivity and losses do not apply to the metrics returned by the traceroute tool; therefore, in order to compare results of traceroute measurements, the only possibility is to add to the stored results a specification of the operating system and version for the traceroute tool used. This document, in order to store results of traceroute measurements and allow comparison of them, defines a standard way to store them using a XML schema. The document is organized as follows: Section 2 defines the terminology used in this document, Section 3 describes the traceroute tool, Section 4 describes the results of a traceroute measurement as displayed to the screen from which the traceroute tool was launched. Section 5 and Section 6 respectively describe the information model and data model for storing configuration and results of the traceroute measurements. The document ends with security considerations and IANA considerations in Section 8 and Section 9 respectively.

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. Terminology used in this document

The terminology used in this document is defined as follow:

- traceroute tool: a software tool for network diagnostic behaving like described in Section 3;
- traceroute measurement: an instance of the traceroute tool launched, with specific configuration parameters (traceroute

measurement configuration parameters), from a specific host (initiator of the traceroute measurement) giving as output specific traceroute measurement results;
- o traceroute probe: one of many IP packets send out by the traceroute tool during a traceroute measurement;
- o traceroute measurement configuration parameters: the configuration parameters of a traceroute measurement;
- o traceroute measurement results: the results of a traceroute measurement;
- o traceroute measurement information: both the results and the configuration parameters of a traceroute measurement;

- o traceroute measurement path: a sequence of hosts transited in order by traceroute probes during a traceroute measurement;

3. The Traceroute tool and its operations

Traceroute is a network diagnostic tool used to determine the hop by hop path from a source to a destination and the Round Trip Time (RTT) from the source to each hop. Traceroute can be therefore used to discover some information (hop counts, delays, etc.) about the path between the initiator of the traceroute measurement and other hosts.

Typically, the traceroute tool attempts to discover the path to a destination by sending UDP probes with specific time-to-live (TTL) values in the IP packet header and trying to elicit an ICMP TIME_EXCEEDED response from each gateway along the path to some host.

More in detail, a first set of probes with TTL equal to 1 are sent by the traceroute tool from the host initiating the traceroute measurement (some tool implementations allow setting the initial TTL to a value equal to "n" different from 1, so that the first "n-1" hops are skipped and the first hop that will be traced is the "n-th" in the path). Upon receiving a probe, the first hop host decreases the TTL value (by one or more). By observing a TTL value equal to zero, the host rejects the probe and typically returns an ICMP message with a TIME_EXCEEDED value. The traceroute tool can therefore derive the IP address of the first hop from the header of the ICMP message and evaluate the RTT between the host initiating the traceroute measurement and the first hop. The next hops are

discovered following the same procedure, taking care of increasing at each step the TTL value of the probes by one. The TTL value is increased until either an ICMP PORT_UNREACHABLE message is received, meaning that the destination host has been reached, or the maximum configured number of hops has been hit.

Some implementations, use ICMP Echos, instead of UDP datagrams. However, many routers do not return ICMP messages about ICMP messages, i.e. no ICMP TIME_EXCEEDED is returned for an ICMP Echo. Therefore, this document recommends to base implementations on UDP datagrams. Considerations on TCP-based implementations of the traceroute tool are reported in Appendix A.1.

4. Results of traceroute measurements

The following list reports the information fields provided as results by all traceroute tool implementations considered. The order in which they are reported here is not relevant and it changes in

different implementations. For each hop the information reported is:

- the hop index;
- the host symbolic address, provided that at least one of the probes received a response, the symbolic address could be resolved at the corresponding host and that the option to display only numerical addresses was not set;
- the host IP address, provided that at least one of the probes received a response;
- the RTT for each response to a probe.

Depending on the traceroute tool implementation, additional information might be displayed in the output (for instance MPLS-related information).

It might happen that some probes do not receive a response within the configured time-out (for instance if the probe is filtered out by a firewall). In this case, an "*" is displayed in place of the RTT. The information model reflects this using a string with the value of "RoundTripTimeNotAvailable" meaning either the probe was lost because of a time-out or it was not possible to transmit a probe. It may also happen that some implementations print the same line multiple times when a router decreases the TTL by more than one looking like multiple hops, the information model is not impacted by this since

each line is handled separately and it is left to the applications handling the XML file how to deal with it. Moreover, for delays below 1 ms, some implementations reports 0 ms (e.g. UNIX and LINUX) while WINDOWS tracert reports "< 1 ms".

5. Information Model for Traceroute Measurements

The information model is composed of information elements; for defining these information elements, a template is used. Such template is specified in the list below:

- name - A unique and meaningful name for the information element. The preferred spelling for the name is to use mixed case if the name is compound, with an initial lower case letter, e.g., "sourceIpAddress".
- description - The semantics of this information element.
- dataType - One of the types listed in Section 5.1 of this document or in an extension of the information model. The type space for attributes is constrained to facilitate implementation.
- units - If the element is a measure of some kind, the units identify what the measure is.

5.1. Data Types

This section describes the set of valid data types of the information model.

- String - The type "String" represents a finite length string of valid characters from the Unicode character encoding set. Unicode allows for ASCII and many other international character sets to be used. It is expected that strings will be encoded in UTF-8 format, which is identical in encoding for USASCII characters, but also accommodates other Unicode multi-byte characters.
- InetAddressType - The type "InetAddressType" represents a type of Internet address. The allowed values are to be intended as imported from [RFC4001] (where the intent was to import only some of the values); additional allowed value are "asnumber" and

"noSpecification".

- InetAddress - The type "InetAddress" denotes a generic Internet address. The allowed values are to be intended as imported from [RFC4001], where the intent was to import only some of the values; an additional allowed value is the AS number to be indicated as the actual number plus the indication how the mapping from IP address to AS number was performed.
- ipASNumberMappingType - The type "ipASNumberMappingType" represents a type of mapping from IP to AS number, it indicated the method that was used to do get the mapping (allowed values are "bgptables", "routingregistries", "nslookup", "others" or "unknown".
- TruthValue - The type "TruthValue" represents a Boolean value. The allowed values are to be intended as imported from [RFC2579].
- Unsigned32 - The type "Unsigned32" represents a value in the range (0..4294967295).
- Unsigned16 - The type "Unsigned16" represents a value in the range (0..65535).
- Unsigned8 - The type "Unsigned8" represents a value in the range (0..255).
- InterfaceIndexOrZero - The type "InterfaceIndexOrZero" is an extension of the InterfaceIndex convention. The latter defines a greater than zero value used to identify an interface or interface sub-layer in the system. This extension permits the additional value of zero. Examples of the usage of zero might include situations where interface was unknown, or when none or all interfaces need to be referenced. The allowed values are to be intended as imported from [RFC2863].
- ProbesType - The type "ProbesType" represents a way of indicating the protocol used for the traceroute probes. Allowed values are UDP, TCP, ICMP.

- DateAndTime - The type "DateAndTime" represents a date-time specification. The allowed values are to be intended as imported from [RFC2579] apart from the fact that in this document there is the need to use a millisecond resolution instead a decisecond one.
- OperationResponseStatus - The type "OperationResponseStatus" is used to report the result of an operation. The allowed values are to be intended as imported from [RFC4560].

5.2. Information Elements

This section describes the elements related to the storing of a traceroute measurement. The elements are grouped in two groups (Configuration and Results) according to their semantics. In order to relate configuration and results elements by means of a common unique identifier, an additional element is defined belonging to both the two groups.

5.2.1. Relationship between the Information Elements

Every traceroute measurement is represented by an instance of the "traceRoute" element. This class provides a standardized representation for traceroute measurement data. The "traceroute" element is an element that can be composed of (depending on the nature of the traceroute measurement):

- 1 "Request" element - containing the traceroute measurement configuration parameters;
- 1 "MeasurementMetadata" element - containing the traceroute measurement parameters actually used;
- 1 "Measurement" element - containing the traceroute measurement results;
- 1 "Request" element and 1 "MeasurementMetadata" element;
- 1 "Request" element and 1 "Measurement" element;
- 1 "Measurement" element and 1 "MeasurementMetadata" element;
- 1 "Request" element, 1 "MeasurementMetadata" element and 1 "Measurement" element;

The "Request" element (as well as the "MeasurementMetadata" element, since they are of the same "_Metadata" type) is a sequence that contains:

- 1 "TestName" element;
- 1 optional "OSName" element;
- 1 optional "OSVersion" element;
- 1 optional "ToolVersion" element;
- 1 optional "ToolName" element;

- 1 "CtlTargetAddressType" element;

- 1 "CtlTargetAddress" element;
- 1 "CtlBypassRouteTable" element;
- 1 "CtlProbeDataSize" element;
- 1 "CtlTimeOut" element;
- 1 "CtlProbesPerHop" element;
- 1 "CtlPort" element;
- 1 "CtlMaxTtl" element;
- 1 "CtlDSField" element;
- 1 "CtlSourceAddressType" element;
- 1 "CtlSourceAddress" element;
- 1 optional "CtlIfIndex" element;
- 1 optional "CtlMiscOptions" element;
- 1 optional "CtlMaxFailures" element;
- 1 "CtlDontFragment" element;
- 1 "CtlInitialTtl" element;
- 1 optional "CtlDescr" element;
- 1 "CtlType" element;

Configuration Information Elements (both "Request" and "MeasurementMetadata" elements) can describe not just traceroute measurements that have already happened ("MeasurementMetadata" elements), but also configuration to be used when requesting a measurement to be made ("Request" element). This is quite different semantically, even if the individual information elements are the same since the elements contained are exactly the same.

The "Measurement" element ("_Measurement" type) is a sequence that contains:

- 1 "TestName" element;
- 1 "ResultsStartDateAndTime" element;
- 1 "ResultsIpTgtAddrType" element;
- 1 "ResultsIpTgtAddr" element;
- 1..2550 "ResultsProbe" elements;
- 0..255 "ResultsHopRawOutputData" elements;
- 1 "ResultsEndDateAndTime" element;

Additionally it is important to say that each "ResultsProbe" element is a sequence that ontains:

- 1 "Index" element;
- 1 "HopIndex" element;
- 1 "IndexPerHop" element;
- 1 "HopAddrType" element;
- 1 "HopAddr" element;

- 1 optional "HopGeoLocation" element;
- 1..255 optional "MPLSLabelStackEntry" elements;
- 1 "RoundTripTime" element;
- 1 "ResponseStatus" element;
- 1 "Time" element;

5.2.2. Configuration Information Elements

This section describes the elements specific to the configuration of the traceroute measurement.

5.2.2.1. CtlTargetAddressType

- name - CtlTargetAddressType
- description - Specifies the type of destination address used in the traceroute measurement.
- dataType - InetAddressType
- units - N/A

5.2.2.2. CtlTargetAddress

- name - CtlTargetAddress
- description - Specifies the host address used in the traceroute measurement. The host address type can be determined by the examining the value of the corresponding CtlTargetAddressType.
- dataType - InetAddress
- units - N/A

5.2.2.3. CtlBypassRouteTable

- name - CtlBypassRouteTable
- description - Specifies if the optional bypassing of the route table was enabled or not. If enabled, the normal routing tables will be bypassed and the probes will be sent directly to a host on an attached network. If the host is not on a directly-attached network, an error is returned. This option can be used to perform the traceroute measurement to a local host through an interface that has no route defined. Please refer to "SO_DONTROUTE" for more explanations regarding this.
- dataType - TruthValue
- units - N/A

5.2.2.4. CtlProbeDataSize

- name - CtlProbeDataSize
- description - Specifies the size of the probes of a traceroute

measurement in octets. If UDP datagrams are used as probes, then the value contained in this object is exact. If another protocol

is used to transmit probes (i.e. TCP or ICMP) for which the specified size is not appropriate, then the implementation can use whatever size (appropriate to the method) is closest to the specified size. The maximum value for this object was computed by subtracting the smallest possible IP header size of 20 octets (IPv4 header with no options) and the UDP header size of 8 octets from the maximum IP packet size. An IP packet has a maximum size of 65535 octets (excluding IPv6 Jumbograms).

- dataType - Unsigned32
- units - octets

5.2.2.5. CtlTimeOut

- name - CtlTimeOut
- description - Specifies the time-out value, in seconds, for each probe of a traceroute measurement.
- dataType - Unsigned32
- units - seconds

5.2.2.6. CtlProbesPerHop

- name - CtlProbesPerHop
- description - Specifies the number of probes with the same time-to-live (TTL) value that are sent for each host.
- dataType - Unsigned32
- units - probes

5.2.2.7. CtlPort

- name - CtlPort
- description - Specifies the base port used by the traceroute measurement.
- dataType - Unsigned16
- units - Port number

5.2.2.8. CtlMaxTtl

- name - CtlMaxTtl
- description - Specifies the maximum TTL value for the traceroute

measurement.
- dataType - Unsigned8
- units - time-to-live value

5.2.2.9. CtlDSField

- name - CtlDSField

- description - Specifies the value that was stored in the Differentiated Services (DS) field in the traceroute probe. The DS Field is defined as the Type of Service (TOS) octet in a IPv4 header or as the Traffic Class octet in a IPv6 header. The value of this object must be a decimal integer in the range from 0 to 255. This option can be used to determine what effect an explicit DS field setting has on a traceroute measurement and its probes. Not all values are legal or meaningful. Useful TOS octet values are probably '16' (low delay) and '8' (high throughput). Further references can be found in [RFC2474] for the definition of the Differentiated Services (DS) field and to [RFC1812] Section 5.3.2 for Type of Service (TOS).
- dataType - Unsigned8
- units - N/A

5.2.2.10. CtlSourceAddressType

- name - CtlSourceAddressType
- description - Specifies the type of the source address, CtlSourceAddress, used in the traceroute measurement.
- dataType - InetAddressType
- units - N/A

5.2.2.11. CtlSourceAddress

- name - CtlSourceAddress
- description - Specifies the IP address (which has to be given as an IP number, not a hostname) as the source address used in traceroute probes. On hosts with more than one IP address, this option can be used to force the source address to be something other than the primary IP address of the interface the probe is sent on. A zero length octet string value for this object means

that source address specification was disabled. The address type (InetAddressType) that relates to this object is specified by the corresponding value of CtlSourceAddressType.
- dataType - InetAddress
- units - N/A

5.2.2.12. CtlIfIndex

- name - CtlIfIndex
- description - Specifies the interface index used in the traceroute measurement for sending the traceroute probes. A value of zero for this object implies that the interface was unknown.
- dataType - InterfaceIndexOrZero
- units - N/A

5.2.2.13. CtlMiscOptions

- name - CtlMiscOptions
- description - Specifies implementation dependent options.
- dataType - String
- units - N/A

5.2.2.14. CtlMaxFailures

- name - CtlMaxFailures
- description - Specifies the maximum number of consecutive timeouts allowed before terminating a traceroute measurement. A value of either 255 (maximum hop count/possible TTL value) or a 0 indicates that the function of terminating a remote traceroute measurement when a specific number of consecutive timeouts are detected was disabled. This element is included to give full compatibility with [RFC4560]. No known implementation of traceroute currently supports it.
- dataType - Unsigned8
- units - timeouts

5.2.2.15. CtlDontFragment

- name - CtlDontFragment
- description - Specifies if the don't fragment flag (DF) in the IP

header for a probe was enabled or not. Setting the DF flag can be used for performing a manual PATH MTU test.
- dataType - TruthValue
- units - N/A

5.2.2.16. CtlInitialTtl

- name - CtlInitialTtl
- description - Specifies the initial TTL value used in a traceroute measurement. Such TTL setting is intended to bypass the initial (often well known) portion of a path.
- dataType - Unsigned8
- units - N/A

5.2.2.17. CtlDescr

- name - CtlDescr
- description - The purpose of this element is to provide a description of the traceroute measurement.
- dataType - String
- units - N/A

5.2.2.18. CtlType

- name - CtlType
- description - Specifies the implementation method used for the traceroute measurement. It specifies if the traceroute is using TCP, UDP, ICMP or others type of probes.
- dataType - ProbesType
- units - N/A

5.2.3. Results Information Elements

This section describes the elements specific to the results of the traceroute measurement.

5.2.3.1. ResultsStartDateAndTime

- name - ResultsStartDateAndTime
- description - Specifies the date and start time of the traceroute

measurement. This is the time when the first probe was seen at the sending interface.
- dataType - DateAndTime
- units - N/A

5.2.3.2. ResultsIpTgtAddrType

- name - ResultsIpTgtAddrType
- description - Specifies the type of address stored in the corresponding ResultsIpTgtAddr element.
- dataType - InetAddressType
- units - N/A

5.2.3.3. ResultsIpTgtAddr

- name - ResultsIpTgtAddr
- description - Specifies the IP address associated with a CtlTargetAddress value when the destination address is specified as a DNS name. The value of this object should be a zero length octet string when a DNS name is not specified or when a specified DNS name fails to resolve.
- dataType - InetAddress
- units - N/A

5.2.3.4. Index

- name - Index
- description - Specifies an index that consecutively numbers all probes for which a reply was received in the sequential order in which the replies were received. The maximum value for this

object is CtlMaxTtl*CtlProbesPerHop.
- dataType - Unsigned32
- units - N/A

5.2.3.5. HopIndex

- name - HopIndex
- description - Specifies which hop in a traceroute measurement path the probe's results are for.
- dataType - Unsigned8
- units - N/A

5.2.3.6. IndexPerHop

- name - IndexPerHop
- description - Specifies the index of a probe for a particular hop in a traceroute measurement path. The number of probes per hop is determined by the value of the corresponding CtlProbesPerHop element.
- dataType - Unsigned8
- units - N/A

5.2.3.7. HopAddrType

- name - HopAddrType
- description - Specifies the type of address stored in the corresponding HopAddr element.
- dataType - InetAddressType
- units - N/A

5.2.3.8. HopAddr

- name - HopAddr
- description - Specifies the address of a hop in the traceroute measurement path. This object is not allowed to be a DNS name. The value of the corresponding object, HopAddrType, indicates this object's IP address type.
- dataType - InetAddress
- units - N/A

5.2.3.9. HopGeoLocation

- name - HopGeoLocation
- description - Specifies the geo location of a hop in the traceroute measurement path represented according to [RFC3825].
- dataType - String

- units - N/A

5.2.3.10. MPLSLabelStackEntry

- name - MPLSLabelStackEntry
- description - Specifies entries of the MPLS label stack of a probe observed when the probe arrived at the hop that replied to the probe. This object contains one MPLS label stack entry as 32 bit value as it is observed on the MPLS label stack. Contained in this single number are the MPLS label, the Exp field, the S flag, and the MPLS TTL value as specified in RFC 3032. If more than one MPLS label stack antry is reported then multiple instances of elements of this type are used. They must be ordered in the same order as on the label stack with the top label stack entry being reported first.
- dataType - Unsigned32
- units - N/A

5.2.3.11. RoundTripTime

- name - RoundTripTime
- description - Specifies the amount of time measured in milliseconds from when a probe was sent to when its response was received or when it timed out. The value of this element is reported as the truncation of the number reported by the traceroute tool (the output "< 1 ms" is therefore encoded as 0 ms). A string with the value of "RoundTripTimeNotAvailable" means either the probe was lost because of a timeout or it was not possible to transmit a probe.
- dataType - Unsigned32 or String
- units - milliseconds or N/A

5.2.3.12. ResponseStatus

- name - ResponseStatus
- description - Specifies the result of a traceroute measurement made by the host for a particular probe.
- dataType - OperationResponseStatus
- units - N/A

5.2.3.13. Time

- name - Time
- description - Specifies the timestamp for when the response to the probe was received at the interface.
- dataType - DateAndTime

- units - N/A

5.2.3.14. ResultsHopRawOutputData

- name - ResultsHopRawOutputData
- description - Specifies the raw output data returned by the traceroute measurement for a certain hop in a traceroute measurement path. It is an implementation-dependant printable string, expected to be useful for a human interpreting the traceroute results.
- dataType - String
- units - N/A

5.2.3.15. ResultsEndDateAndTime

- name - ResultsEndDateAndTime
- description - Specifies the date and end time of the traceroute measurement. It is either the time when the response to the last probe of the traceroute measurement was received or the time when the last probe of the traceroute measurement was sent plus the relative timeout (in case of missing response).
- dataType - DateAndTime
- units - N/A

5.2.4. Information Element Correlating Configuration and Results Elements

This section defines an additional element belonging to both the two previous groups (configuration elements and result elements) named TestName. This element is defined in order to relate configuration elements and results ones by means of a common unique identifier (to be considered unique within the scope of a specific host, initiator of the traceroute measurement).

5.2.4.1. TestName

- name - TestName
- description - Specifies the name of a traceroute measurement. This is locally unique, within the scope of a specific host, initiator of the traceroute measurement.
- dataType - String
- units - N/A

5.2.5. Information Elements to compare traceroute measurements results one with each other

This section defines additional elements belonging to both the two

previous groups (configuration elements and result elements); these

elements were defined in order to allow traceroute measurements results comparison among different traceroute measurements.

5.2.5.1. OSName

- name - OSName
- description - Specifies the name of the operating system on which the traceroute measurement was launched.
- dataType - String
- units - N/A

5.2.5.2. OSVersion

- name - OSVersion
- description - Specifies the OS version on which the traceroute measurement was launched.
- dataType - String
- units - N/A

5.2.5.3. ToolVersion

- name - ToolVersion
- description - Specifies the version of the traceroute tool used.
- dataType - String
- units - N/A

5.2.5.4. ToolName

- name - ToolName
- description - Specifies the name of the traceroute tool used.
- dataType - String
- units - N/A

6. Data Model for Storing Traceroute Measurements

For storing and transmitting information according to the information model defined in the previous section, a data model is required that specifies how to encode the elements of the information model.

There are several design choices for a data model. It can use a binary or textual representation and it can be defined from scratch or use already existing frameworks and data models. In general, the use of already existing frameworks and models should be preferred.

Binary and textual representation both have advantages and disadvantages. Textual representations are (with some limitations) human readable while a binary representation consumes less resources

for storing, transmitting and parsing data.

An already existing and closely related data model is the DISMAN-TRACEROUTE-MIB module [RFC4560], that specifies a BER encoding [RFC3417] used by the Simple Network Management Protocol (SNMP) [RFC3410] for transmitting traceroute measurement information (configuration and results). This data model is well suited and supported within network management systems, but as a general format for storing and transmitting traceroute results it is not easily applicable.

Another binary representation would be an extension of traffic flow information encodings as specified for the IPFIX protocol [RFC5101], [RFC5102]. The IPFIX protocol is extensible. However, the architecture behind this protocol [I-D.ietf-ipfix-architecture] is targeted at exporting passively measured flow information. Therefore, some obstacles are expected when trying to use it for transmitting traceroute measurements information.

For textual representations, using the eXtensible Markup Language (XML) [XML] is an obvious choice. XML supports clean structuring of data and syntax checking of records. With some limitations it is human readable. It is supported well by a huge pool of tools and standards for generating, transmitting, parsing and converting it to other data formats. Its disadvantages is the resource consumption for processing, storing, and transmitting information. Since the expected data volumes related to traceroute measurements in network operation and maintenance is not expected to be extremely high, the inefficient usage of resources is not a significant disadvantage. Therefore, XML was chosen as basis for the traceroute measurements information model that is specified in this section.

Section 7 contains the XML schema to be used as a template for

storing and/or exchanging traceroute measurements information. The schema was designed in order to use an extensible approach based on templates (pretty similar to how IPFIX protocol is designed) where the traceroute configuration elements (both the requested parameters, Request, and the actual parameters used, MeasurementMetadata) are metadata to be referenced by results information elements (data) by means of the TestName element (used as unique identifier, within the scope of a specific host, initiator of the traceroute measurement). Currently Open Grid Forum (OGF) is also using this approach and cross-requirements have been analyzed. As a result of this analysis the XML schema contained in Section 7 is compatible with OGF schema since it was designed in a way that both limits the unnecessary redundancy and a simple one-to-one transformation between the two exist.

7. XML Schema for traceroute Measurements

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="urn:ietf:params:xml:ns:traceroute-1.0">

  <xs:simpleType name="inetAddressType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="unknown"/>
      <xs:enumeration value="ipv4"/>
      <xs:enumeration value="ipv6"/>
      <xs:enumeration value="dns"/>
      <xs:enumeration value="asnumber"/>
      <xs:enumeration value="noSpecification"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="inetAddressTypeWithoutDns">
    <xs:restriction base="xs:string">
      <xs:enumeration value="unknown"/>
      <xs:enumeration value="ipv4"/>
      <xs:enumeration value="ipv6"/>
      <xs:enumeration value="asnumber"/>
      <xs:enumeration value="noSpecification"/>
    </xs:restriction>
  </xs:simpleType>
```

```
<xs:simpleType name="_zeroLengthString">
  <xs:restriction base="xs:string">
    <xs:maxLength value="0"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="_inetAddressIpv4">
  <xs:restriction base="xs:string">
    <xs:pattern value="(([1-9]?[0-9]|1[0-9][0-9]|
    2[0-4][0-9]|25[0-5]).){3}([1-9]?[0-9]|1[0-9]
    [0-9]|2[0-4][0-9]|25[0-5])"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="_inetAddressIpv6">
  <xs:restriction base="xs:string">
    <xs:pattern value="(([\dA-Fa-f]{1,4}:){7}[\dA-Fa-f]{1,4})
    (:([\d]{1,3}.){3}[\d]{1,3})?"/>
  </xs:restriction>
</xs:simpleType>
```



```
<xs:simpleType name="_inetAddressDns">
  <xs:restriction base="xs:string">
    <xs:maxLength value="256"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="_asNumber">
  <xs:restriction base="xs:unsignedInt"/>
</xs:simpleType>

<xs:simpleType name="_ipASNumberMappingType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="bgptables"/>
    <xs:enumeration value="routingregistries"/>
    <xs:enumeration value="nslookup"/>
    <xs:enumeration value="others"/>
    <xs:enumeration value="unknown"/>
  </xs:restriction>
</xs:simpleType>
```

```
    <xs:simpleType name="operationResponseStatus">
      <xs:restriction base="xs:string">
        <xs:enumeration value="responseReceived"/>
        <xs:enumeration value="unknown"/>
        <xs:enumeration value="internalError"/>
        <xs:enumeration value="requestTimedOut"/>
        <xs:enumeration value="unknownDestinationAddress"/>
        <xs:enumeration value="noRouteToTarget"/>
        <xs:enumeration value="interfaceInactiveToTarget"/>
        <xs:enumeration value="arpFailure"/>
        <xs:enumeration value="maxConcurrentLimitReached"/>
        <xs:enumeration value="unableToResolveDnsName"/>
        <xs:enumeration value="invalidHostAddress"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_dateAndTime">
      <xs:restriction base="xs:dateTime"/>
    </xs:simpleType>

    <xs:simpleType name="_TestName">
      <xs:annotation>
        <xs:documentation>Specifies the name of a
        traceroute measurement. This is locally unique,
        within the scope of a specific host, initiator of
        the traceroute measurement.
        </xs:documentation>
      </xs:annotation>
```



```
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_OSName">
      <xs:annotation>
        <xs:documentation>Specifies the name of the operating
        system on which the traceroute measurement was
        launched.
        </xs:documentation>
      </xs:annotation>
```

```
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_OSVersion">
      <xs:annotation>
        <xs:documentation>Specifies the OS version on which the
        traceroute measurement was launched.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_ToolVersion">
      <xs:annotation>
        <xs:documentation>Specifies the version of the traceroute
        tool used.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_ToolName">
      <xs:annotation>
        <xs:documentation>Specifies the name of the traceroute
        tool used.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
```



```
    </xs:simpleType>

    <xs:simpleType name="_CtlBypassRouteTable">
      <xs:annotation>
        <xs:documentation>Specifies if the optional bypassing
```

```
      of the route table was enabled or not. If enabled,
      the normal routing tables will be bypassed and the
      probes will be sent directly to a host on an attached
      network. If the host is not on a directly-attached
      network, an error is returned.  This option can be
      used to perform the traceroute measurement to a
      local host through an interface that has no route
      defined. Please refer to SO_DONTROUTE for
      more explanations regarding this.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:boolean"/>
  </xs:simpleType>

  <xs:simpleType name="_CtlProbeDataSize">
    <xs:annotation>
      <xs:documentation>Specifies the size of the probes
      of a traceroute measurement in octets. If UDP
      datagrams are used as probes, then the value
      contained in this object is exact. If another
      protocol is used to transmit probes (i.e. TCP or
      ICMP) for which the specified size is not
      appropriate, then the implementation can use
      whatever size (appropriate to the method) is
      closest to the specified size. The maximum value
      for this object was computed by subtracting the
      smallest possible IP header size of 20 octets (IPv4
      header with no options) and the UDP header size of
      8 octets from the maximum IP packet size. An IP
      packet has a maximum size of 65535 octets (excluding
      IPv6 Jumbograms).
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:unsignedShort">
      <xs:maxExclusive value="65508"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="_CtlTimeOut">
    <xs:annotation>
      <xs:documentation>Specifies the time-out value, in
      seconds, for each probe of a traceroute measurement.
      </xs:documentation>
```

```
  </xs:annotation>
  <xs:restriction base="xs:unsignedByte">
    <xs:minExclusive value="0"/>
    <xs:maxExclusive value="61"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="_CtlProbesPerHop">
  <xs:annotation>
    <xs:documentation>Specifies the number of probes
    with the same time-to-live (TTL) value that are
    sent for each host.
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:unsignedByte">
    <xs:minExclusive value="0"/>
    <xs:maxExclusive value="11"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="_CtlPort">
  <xs:annotation>
    <xs:documentation>Specifies the base port used
    by the traceroute measurement.
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:unsignedByte">
    <xs:minExclusive value="0"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="_CtlMaxTtl">
  <xs:annotation>
    <xs:documentation>Specifies the maximum TTL value
    for the traceroute measurement.
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:unsignedByte">
    <xs:minExclusive value="0"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="_CtlDSField">
  <xs:annotation>
    <xs:documentation>Specifies the value that was
    stored in the Differentiated Services (DS) field
    in the traceroute probe. The DS Field is defined
    as the Type of Service (TOS) octet in a IPv4 header
```

```
          or as the Traffic Class octet in a IPv6 header.
          The value of this object must be a decimal integer
          in the range from 0 to 255. This option can be
          used to determine what effect an explicit DS field
          setting has on a traceroute measurement and its
          probes. Not all values are legal or meaningful.
          Useful TOS octet values are probably '16' (low
          delay) and '8' (high throughput). Further references
          can be found in RFC 2474 for the definition of the
          Differentiated Services (DS) field and to RFC 1812
          Section 5.3.2 for Type of Service (TOS).
          </xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:unsignedByte"/>
      </xs:simpleType>

      <xs:simpleType name="_CtlIfIndex">
        <xs:annotation>
          <xs:documentation>Specifies the interface index
          used in the traceroute measurement for sending
          the traceroute probes. A value of zero for this
          object implies that the interface was unknown.
          </xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:unsignedByte"/>
      </xs:simpleType>

      <xs:simpleType name="_CtlMiscOptions">
        <xs:annotation>
          <xs:documentation>Specifies implementation dependent
          options.</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
          <xs:maxLength value="255"/>
        </xs:restriction>
      </xs:simpleType>

      <xs:simpleType name="_CtlMaxFailures">
        <xs:annotation>
          <xs:documentation>Specifies the maximum number
          of consecutive timeouts allowed before terminating
          a traceroute measurement. A value of either 255
```

```
        (maximum hop count/possible TTL value) or a 0
        indicates that the function of terminating a
        remote traceroute measurement when a specific
        number of consecutive timeouts are detected was
        disabled. This element is included to give full
        compatibility with RFC 4560. No known implementation
```



```
        of traceroute currently supports it.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:unsignedByte"/>
    </xs:simpleType>

    <xs:simpleType name="_CtlDontFragment">
      <xs:annotation>
        <xs:documentation>Specifies if the don't fragment
        flag (DF) in the IP header for a probe was enabled
        or not. Setting the DF flag can be used for
        performing a manual PATH MTU test.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:boolean"/>
    </xs:simpleType>

    <xs:simpleType name="_CtlInitialTtl">
      <xs:annotation>
        <xs:documentation>Specifies the initial TTL
        value used in a traceroute measurement. Such
        TTL setting is intended to bypass the initial
        (often well known) portion of a path.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:unsignedByte">
        <xs:minExclusive value="0"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_CtlDescr">
      <xs:annotation>
        <xs:documentation>The purpose of this element
        is to provide a description of the traceroute
        measurement.
```

```
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_CtlType">
      <xs:annotation>
        <xs:documentation>Specifies the implementation
        method used for the traceroute measurement.
        It specifies if the traceroute is using TCP,
        UDP, ICMP or others type of probes.
```



```
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:enumeration value="TCP"/>
        <xs:enumeration value="UDP"/>
        <xs:enumeration value="ICMP"/>
        <xs:enumeration value="others"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_Index">
      <xs:annotation>
        <xs:documentation>Specifies an index that
        consecutively numbers all probes for which
        a reply was received in the sequential order
        in which the replies were received. The
        maximum value for this object is
        CtlMaxTtl*CtlProbesPerHop.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:unsignedShort">
        <xs:minExclusive value="0"/>
        <xs:maxExclusive value="2551"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_HopIndex">
      <xs:annotation>
```

```
        <xs:documentation>Specifies which hop in a
        traceroute measurement path the probe's
        results are for.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:unsignedByte">
        <xs:minExclusive value="0"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_IndexPerHop">
      <xs:annotation>
        <xs:documentation>Specifies the index of a
        probe for a particular hop in a traceroute
        measurement path. The number of probes per hop
        is determined by the value of the corresponding
        CtlProbesPerHop element.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:unsignedByte">
```



```
        <xs:minExclusive value="0"/>
        <xs:maxExclusive value="11"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_HopGeoLocation">
      <xs:annotation>
        <xs:documentation>Specifies the geo location of a
        hop in the traceroute measurement path represented
        according to RFC 3825.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
    </xs:simpleType>

   <xs:simpleType name="_MPLSLabelStackEntry">
      <xs:annotation>
        <xs:documentation>Specifies entries of the
        MPLS label stack of a probe observed when the probe
```

```
        arrived at the hop that replied to the probe. This
        object contains one MPLS label stack entry as 32 bit
        value as it is observed on the MPLS label stack.
        Contained in this single number are the MPLS label,
        the Exp field, the S flag, and the MPLS TTL value as
        specified in RFC 3032.
        If more than one MPLS label stack antry is reported
        then multiple instances of elements of this type are
        used. They must be ordered in the same order as on
        the label stack with the top label stack entry being
        reported first.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:unsignedInt">
        <xs:maxInclusive value="4294967295"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_probeRoundTripTime">
      <xs:restriction base="xs:unsignedShort">
        <xs:maxExclusive value="60001"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_probeRoundTripTimeNotAvailable">
      <xs:restriction base="xs:string">
        <xs:enumeration value="NotAvailable"/>
```



```
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="_ResultsHopRawOutputData">
      <xs:annotation>
        <xs:documentation>Specifies the raw output data
        returned by the traceroute measurement for a certain
        hop in a traceroute measurement path. It is
        an implementation-dependant printable string,
        expected to be useful for a human interpreting the
        traceroute results.
        </xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
```

```
      <xs:maxLength value="200"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="_inetAddressASNumber">
    <xs:annotation>
      <xs:documentation>Specifies the AS number of a hop in the
      traceroute path as a 32 bit number and the indication how
      the mapping from IP address to AS number was performed.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="asNumber"
                  type="_asNumber"/>
      <xs:element name="ipASNumberMappingType"
                  type="_ipASNumberMappingType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="_inetAddress">
    <xs:choice>
      <xs:element name="inetAddressUnknown"
                  type="_zeroLengthString"/>
      <xs:element name="inetAddressIpv4"
                  type="_inetAddressIpv4"/>
      <xs:element name="inetAddressIpv6"
                  type="_inetAddressIpv6"/>
      <xs:element name="inetAddressDns"
                  type="_inetAddressDns"/>
      <xs:element name="inetAddressASNumber"
                  type="_inetAddressASNumber"/>
    </xs:choice>
  </xs:complexType>
```



```
  <xs:complexType name="_inetAddressWithoutDns">
    <xs:choice>
      <xs:element name="inetAddressUnknown"
                  type="_zeroLengthString"/>
      <xs:element name="inetAddressIpv4"
                  type="_inetAddressIpv4"/>
      <xs:element name="inetAddressIpv6"
```

```
                    type="_inetAddressIpv6"/>
        <xs:element name="inetAddressASNumber"
                    type="_inetAddressASNumber"/>
        <xs:element name="zeroLengthString"
                    type="_zeroLengthString"/>
      </xs:choice>
    </xs:complexType>

    <xs:complexType name="_CtlTargetAddressType">
      <xs:annotation>
        <xs:documentation>Specifies the type of destination
        address used in the traceroute measurement.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="targetAddressType"
                    type="inetAddressType"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_CtlTargetAddress">
      <xs:annotation>
        <xs:documentation>Specifies the host address
        used in the traceroute measurement. The host
        address type can be determined by the examining
        the value of the corresponding CtlTargetAddressType.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="targetAddress" type="_inetAddress"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_CtlSourceAddressType">
      <xs:annotation>
        <xs:documentation>Specifies the type of the source
        address, CtlSourceAddress, used in the traceroute
        measurement.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
```

```
    <xs:element name="sourceAddressType"
                type="inetAddressTypeWithoutDns"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="_CtlSourceAddress">
  <xs:annotation>
    <xs:documentation>Specifies the IP address (which
    has to be given as an IP number, not a hostname)
    as the source address used in traceroute probes.
    On hosts with more than one IP address, this option
    can be used to force the source address to be
    something other than the primary IP address of the
    interface the probe is sent on. A zero length
    octet string value for this object means that
    source address specification was disabled. The
    address type (InetAddressType) that relates to
    this object is specified by the corresponding
    value of CtlSourceAddressType.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="sourceAddress"
                type="_inetAddressWithoutDns"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="_ResultsStartDateAndTime">
  <xs:annotation>
    <xs:documentation>Specifies the date and start
    time of the traceroute measurement. This is the
    time when the first probe was seen at the sending
    interface.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="dateAndTime" type="_dateAndTime"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="_ResultsIpTgtAddrType">
  <xs:annotation>
    <xs:documentation>Specifies the type of address stored
    in the corresponding ResultsIpTgtAddr element.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="ipTgtAddrType"
```

```
                    type="inetAddressTypeWithoutDns"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_ResultsIpTgtAddr">
      <xs:annotation>
        <xs:documentation>Specifies the IP address associated
        with a CtlTargetAddress value when the destination
        address is specified as a DNS name. The value of
        this object should be a zero length octet string
        when a DNS name is not specified or when a specified
        DNS name fails to resolve.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="ipTgtAddr"
                    type="_inetAddressWithoutDns"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_HopAddrType">
      <xs:annotation>
        <xs:documentation>Specifies the type of address stored
        in the corresponding HopAddr element.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="probeHopAddrType"
                    type="inetAddressTypeWithoutDns"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_HopAddr">
      <xs:annotation>
        <xs:documentation>Specifies the address of a
        hop in the traceroute measurement path. This
        object is not allowed to be a DNS name. The
        value of the corresponding object, HopAddrType,
        indicates this object's IP address type.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="probeHopAddr"
```

```
                    type="_inetAddressWithoutDns"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_RoundTripTime">
```



```
      <xs:annotation>
        <xs:documentation>Specifies the amount of
        time measured in milliseconds from when a
        probe was sent to when its response was
        received or when it timed out. The value of
        this element is reported as the truncation
        of the number reported by the traceroute
        tool (the output "&lt; 1 ms" is therefore
        encoded as 0 ms). A string with the value of
        "RoundTripTimeNotAvailable" means either the
        probe was lost because of a timeout or it
        was not possible to transmit a probe.
        </xs:documentation>
      </xs:annotation>
      <xs:choice>
          <xs:element name="probeRoundTripTime"
                      type="_probeRoundTripTime"/>
          <xs:element name="probeRoundTripTimeNotAvailable"
                      type="_probeRoundTripTime"/>
      </xs:choice>
    </xs:complexType>

    <xs:complexType name="_ResponseStatus">
      <xs:annotation>
        <xs:documentation>Specifies the result of a traceroute
        measurement made by the host for a particular probe.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="probeResponseStatus"
                    type="operationResponseStatus"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_Time">
      <xs:annotation>
```

```
        <xs:documentation>Specifies the timestamp for
        when the response to the probe was received at the
        interface.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="dateAndTime" type="_dateAndTime"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_ResultsProbe">
      <xs:sequence>
```



```
        <xs:element name="Index"
                    type="_Index"/>
        <xs:element name="HopIndex"
                    type="_HopIndex"/>
        <xs:element name="IndexPerHop"
                    type="_IndexPerHop"/>
        <xs:element name="HopAddrType"
                    type="_HopAddrType"/>
        <xs:element name="HopAddr"
                    type="_HopAddr"/>
        <xs:element name="HopGeoLocation"
                    type="_HopGeoLocation"
                    minOccurs="0" maxOccurs="1"/>
        <xs:element name="MPLSLabelStackEntry"
                    type="_MPLSLabelStackEntry"
                    minOccurs="0" maxOccurs="255"/>
        <xs:element name="RoundTripTime"
                    type="_RoundTripTime"/>
        <xs:element name="ResponseStatus"
                    type="_ResponseStatus"/>
        <xs:element name="Time"
                    type="_Time"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_ResultsEndDateAndTime">
      <xs:annotation>
        <xs:documentation>Specifies the date and end time
        of the traceroute measurement. It is either the
```

```
        time when the response to the last probe of the
        traceroute measurement was received or the time
        when the last probe of the traceroute measurement
        was sent plus the relative timeout (in case of
        missing response).
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="dateAndTime" type="_dateAndTime"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_Metadata">
      <xs:annotation>
        <xs:documentation>Specifies the metadata for a
        traceroute operation. In a request, these are the
        requested parameters. In a response, they are the
        actual parameters used.
        </xs:documentation>
```



```
      </xs:annotation>
      <xs:sequence>
        <xs:element name="TestName"
                    type="_TestName"/>
        <xs:element name="OSName"
                    type="_OSName"
                    minOccurs="0" maxOccurs="1"/>
        <xs:element name="OSVersion"
                    type="_OSVersion"
                    minOccurs="0" maxOccurs="1"/>
        <xs:element name="ToolVersion"
                    type="_ToolVersion"
                    minOccurs="0" maxOccurs="1"/>
        <xs:element name="ToolName"
                    type="_ToolName"
                    minOccurs="0" maxOccurs="1"/>
        <xs:element name="CtlTargetAddressType"
                    type="_CtlTargetAddressType"/>
        <xs:element name="CtlTargetAddress"
                    type="_CtlTargetAddress"/>
        <xs:element name="CtlBypassRouteTable"
                    type="_CtlBypassRouteTable"
```

```
                    />
          <xs:element name="CtlProbeDataSize"
                      type="_CtlProbeDataSize"/>
          <xs:element name="CtlTimeOut"
                      type="_CtlTimeOut"/>
          <xs:element name="CtlProbesPerHop"
                      type="_CtlProbesPerHop"/>
          <xs:element name="CtlPort"
                      type="_CtlPort"/>
          <xs:element name="CtlMaxTtl"
                      type="_CtlMaxTtl"/>
          <xs:element name="CtlDSField"
                      type="_CtlDSField"/>
          <xs:element name="CtlSourceAddressType"
                      type="_CtlSourceAddressType"/>
          <xs:element name="CtlSourceAddress"
                      type="_CtlSourceAddress"/>
          <xs:element name="CtlIfIndex"
                      type="_CtlIfIndex" minOccurs="0"
                      maxOccurs="1"/>
          <xs:element name="CtlMiscOptions"
                      type="_CtlMiscOptions" minOccurs="0"
                      maxOccurs="1"/>
          <xs:element name="CtlMaxFailures"
                      type="_CtlMaxFailures" minOccurs="0"
                      maxOccurs="1"/>
```



```
          <xs:element name="CtlDontFragment"
                      type="_CtlDontFragment"/>
          <xs:element name="CtlInitialTtl"
                      type="_CtlInitialTtl"/>
          <xs:element name="CtlDescr"
                      type="_CtlDescr" minOccurs="0"
                      maxOccurs="1"/>
          <xs:element name="CtlType"
                      type="_CtlType"/>
        </xs:sequence>
      </xs:complexType>

      <xs:complexType name="_Measurement">
        <xs:annotation>
          <xs:documentation>
```

```
          Contains the actual traceroute measurement
          results.
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="TestName"
                    type="_TestName"/>
        <xs:element name="ResultsStartDateAndTime"
                    type="_ResultsStartDateAndTime"/>
        <xs:element name="ResultsIpTgtAddrType"
                    type="_ResultsIpTgtAddrType"/>
        <xs:element name="ResultsIpTgtAddr"
                    type="_ResultsIpTgtAddr"/>
        <xs:element name="ResultsProbe"
                    type="_ResultsProbe"
                    minOccurs="1" maxOccurs="2550"/>
        <xs:element name="ResultsHopRawOutputData"
                    type="_ResultsHopRawOutputData"
                    minOccurs="0" maxOccurs="255"/>
        <xs:element name="ResultsEndDateAndTime"
                    type="_ResultsEndDateAndTime"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="_traceRoute">
      <xs:sequence minOccurs="1">
        <xs:element name="Request"
                  type="_Metadata" minOccurs="0" />
        <xs:element name="MeasurementMetadata"
                  type="_Metadata" minOccurs="0" />
        <xs:element name="Measurement"
                  type="_Measurement" minOccurs="0" />
      </xs:sequence>
```



```
    </xs:complexType>

    <!--Reference to "traceRoute" element-->
    <xs:element
     xmlns="urn:ietf:params:xml:ns:traceroute-1.0"
     name="traceRoute" type="_traceRoute"/>

  </xs:schema>
```

8. Security Considerations

Security considerations in this section discuss are grouped into considerations related to conducting traceroute measurements and considerations related to storing and transmitting traceroute measurements information.

This memo does not specify an implementation of a traceroute tool. Neither does it specify a certain procedure for storing traceroute measurements information. Still it is considered desirable to discuss related security issues below.

8.1. Conducting Traceroute Measurements

Conducting Internet measurements can raise both security and privacy concerns. Traceroute measurements, in which traffic is injected into the network, can be abused for denial-of-service attacks disguised as legitimate measurement activity.

Measurement parameters MUST be carefully selected so that the measurements inject trivial amounts of additional traffic into the networks they measure. If they inject "too much" traffic, they can skew the results of the measurement, and in extreme cases cause congestion and denial of service.

The measurements themselves could be harmed by routers giving measurement traffic a different priority than "normal" traffic, or by an attacker injecting artificial measurement traffic. If routers can recognize measurement traffic and treat it separately, the measurements will not reflect actual user traffic. If an attacker injects artificial traffic that is accepted as legitimate, the loss rate will be artificially lowered. Therefore, the measurement methodologies SHOULD include appropriate techniques to reduce the probability measurement traffic can be distinguished from "normal" traffic.

Authentication techniques, such as digital signatures, may be used

where appropriate to guard against injected traffic attacks.

8.2. Securing Traceroute Measurements Information

Traceroute measurement information are not considered highly sensitive. Still, they may contain sensitive information on network paths, routing states, use IP addresses, and roundtrip times, that the operator a networks may want to detect for business or security reasons.

It is thus important to control access to Information acquired by conducting traceroute measurements, particularly when transmitting it over a networks but also when storing it. It is RECOMMENDED that transmission of traceroute measurement information over a network uses appropriate protection mechanisms for preserving privacy, integrity and authenticity. It is further RECOMMENDED that secure authentication and authorization are used for protecting stored traceroute measurement information.

9. IANA Considerations

This document uses URNs to describe an XML namespace and an XML schema for traceroute measurements information storing and transmission conforming to a registry mechanism described in [RFC3688]. Two URI assignments are requested.

1. Registration request for the IPPM traceroute measurements namespace
   * URI: urn:ietf:params:xml:ns:traceroute-1.0
   * Registrant Contact: IESG
   * XML: None. Namespace URIs do not represent an XML
2. Registration request for the IPPM traceroute measurements schema
   * URI: urn:ietf:params:xml:schema:traceroute-1.0
   * Registrant Contact: IESG
   * XML: See the section Section 7 of this document.

10. References

10.1. Normative References

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

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

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

[RFC3825] Polk, J., Schnizlein, J., and M. Linsner, "Dynamic Host Configuration Protocol Option for Coordinate-based Location Configuration Information", RFC 3825, July 2004.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

[RFC4560] Quittek, J. and K. White, "Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations", RFC 4560, June 2006.

10.2. Informative References

[I-D.ietf-ipfix-architecture] Sadasivan, G., "Architecture for IP Flow Information Export", draft-ietf-ipfix-architecture-12 (work in progress), September 2006.

[RFC1812] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

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

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC3417] Presuhn, R., "Transport Mappings for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3417, December 2002.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC5101] Claise, B., "Specification of the IP Flow Information

[RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, January 2008.

[XML] Yergeau et al., F., "Extensible Markup Language (XML) 1.0 (Third Edition)", W3C Recommendation, February 2004.

Appendix A. Traceroute Default Configuration Parameters

This section lists traceroute measurement configuration parameters as well as their defaults on various platforms and illustrates how widely they may vary. This document considered four major traceroute tool implementations and compared them based on configurable parameters and default values. The LINUX (SUSE 9.1), BSD (FreeBSD 7.0) and UNIX (SunOS 5.9) implementations are based on UDP datagrams, while the WINDOWS (XP SP2) one uses ICMP Echos. The comparison is summarized in the following table, where an N/A in the option column, means that such parameter is not configurable for the specific implementation. A comprehensive comparison of available implementations is outside the scope of this document; however, already by sampling a few different implementations, it can be observed that they can differ quite significantly in terms of configurable parameters and also default values.Note that in the following table only those options which are available in at least two of the considered implementations are reported.

OS	Option	Description	Default
LINUX	-m	Specify the maximum TTL used in traceroute probes.	30
FreeBSD	-m		OS var
UNIX	-m		30
WINDOWS	-h		30
LINUX	-n	Display hop addresses	-

FreeBSD	-n	numerically rather than symbolically.	-
UNIX	-n		-
WINDOWS	-d		-
LINUX	-w	Set the time to wait for a response to a probe.	3 sec

FreeBSD	-w		5 sec
UNIX	-w		5 sec
WINDOWS	-w		4 sec
LINUX	N/A	Specify a loose source route gateway (to direct the traceroute probes through routers not necessarily in the path).	-
FreeBSD	-g		-
UNIX	-g		-
WINDOWS	-g		-
LINUX	-p	Set the base UDP port number used in traceroute probes (UDP port = base + nhops - 1).	33434
FreeBSD	-p		33434
UNIX	-p		33434
WINDOWS	N/A		-
LINUX	-q	Set the number of probes per TTL.	3
FreeBSD	-q		3
UNIX	-q		3
WINDOWS	N/A		3
LINUX	-S	Set the IP source address in outgoing probes to the	IP address

FreeBSD	-s	specified value.	of the out interface
UNIX	-s		
WINDOWS	N/A		
LINUX	-t	Set the type-of-service (TOS) in the probes to the specified value.	0
FreeBSD	-t		0
UNIX	-t		0
WINDOWS	N/A		0
LINUX	-v	Verbose output: received ICMP packets other than	-

FreeBSD	-v	TIME_EXCEEDED and UNREACHABLE are listed.	-
UNIX	-v		-
WINDOWS	N/A		-
LINUX	N/A	Set the time (in msec) to pause between probes.	-
FreeBSD	-z		0
UNIX	-P		0
WINDOWS	N/A		-
LINUX	-r	Bypass the normal routing tables and send directly to a host on attached network.	-
FreeBSD	-r		-
UNIX	-r		-
WINDOWS	N/A		-
LINUX	-f	Set the initial TTL for the first probe.	1
FreeBSD	-f		1

UNIX	-f		1
WINDOWS	N/A		1
LINUX	-F	Set the "don't fragment" bit.	-
FreeBSD	-F		-
UNIX	-F		-
WINDOWS	N/A		-
LINUX	N/A	Enables socket level debugging.	-
FreeBSD	-d		-
UNIX	-d		-
WINDOWS	N/A		-
LINUX	N/A	Use ICMP ECHO instead of UDP datagrams.	-

FreeBSD	-I		-
UNIX	-I		-
WINDOWS	N/A		-
LINUX	-I	Specify a network interface to obtain the IP address for outgoing IP packets (alternative to option -s).	-
FreeBSD	-i		-
UNIX	-i		-
WINDOWS	N/A		-
LINUX	N/A	Toggle checksum.	-
FreeBSD	-x		-

UNIX	-x		-
WINDOWS	N/A		-
LINUX	-	As optional last parameter, LINUX, FreeBSD and UNIX implementations allow specifying the probe datagram length for outgoing probes.	Depends on implementation.
FreeBSD	-		
UNIX	-		
WINDOWS	N/A		

A.1. Alternative Traceroute Implementations

As stated above, the widespread use of firewalls might prevent UDP or ICMP based traceroutes to completely trace the path to a destination, since traceroute probes might end up being filtered. In some cases, such limitation might be overcome by sending instead TCP packets to specific ports that hosts located behind the firewall are listening for connections on. TCP based implementations use TCP SYN or FIN probes and listen for TIME_EXCEEDED messages, TCP RESET and other messages from firewalls and gateways on the path. On the other hand, some firewalls filter out TCP SYN packets to prevent denial of service attacks, therefore the actual advantage of using TCP instead of UDP traceroute depends mainly on firewall configurations, which are not known in advance. A detailed analysis of TCP-based traceroute tools and measurements was outside the scope of this document, anyway for completeness reasons the information model supports the storing of TCP-based traceroute measurements, too.

Appendix B. Known Problems with Traceroute

B.1. Compatibility between traceroute measurements results and IPPM metrics

Because of implementation choices, a known inconsistency exists between the round-trip delay metric defined by the IPPM working group in RFC 2681 and the results returned by the current traceroute tool implementations. Unfortunately, it is unlikely that the traceroute tool implementations will implement the standard definition in the near future. The only possibility is therefore to compare results of

different traceroute measurements one with each other; in order to do this, specifications both of the operating system (name and version) and of the traceroute tool version used were added to the metadata elements in order to help in comparing metrics between two different traceroute measurements results (if run using the same operating system and the same version of the tool). Moreover, the traceroute tool has built-in configurable mechanisms like time-outs and can experience problems related to the crossing of firewalls; therefore some of the packets that traceroute sends out end up being time-out or filtered. As a consequence, it might not be possible to trace the path to a node or there might not be a complete set of probes describing the RTT to reach it.

Appendix C. Differences to DISMAN-TRACEROUTE-MIB

For performing remote traceroute operations at managed node, the IETF has standardized the DISMAN-TRACEROUTE-MIB module in [RFC4560]. This module allows:

- retrieving capability information of the traceroute tool implementation at the managed node,
- configuring traceroute measurements to be performed,
- retrieving information about ongoing and completed traceroute measurements,
- retrieving traceroute measurement statistics.

The traceroute storage format described in this document has significant overlaps with this MIB module. Particularly, the models for the traceroute measurement configuration and for the result from completed measurements are almost identical. But for other pats of the DISMAN-TRACEROUTE MIB module there is no need to model them in a traceroute measurements storage format. Particularly, the capability information, information about ongoing measurements and measurement statistics are not covered by the DISMAN traceroute storage model.

Concerning traceroute measurements and their results, there are

structural differences between the two models caused by the different choices for the encoding of the specification. For DISMAN-TRACEROUTE-MIB, the Structure of Management Information (SMIv2, STD 58, RFC 2578 [RFC2578]) was used, while the IPPM traceroute

measurements data model is encoded using XML.

This difference in structure implies that the DISMAN-TRACEROUTE-MIB module contains SMI-specific information element (managed objects) that concern tables of managed objects (specification, entry creation and deletion, status retrieval) that are not required for the XML-encoded traceroute measurements data model.

But for most of the remaining information elements that concern configuration of traceroute measurements and results of completed measurements, the semantics is identical between the DISMAN-TRACEROUTE-MIB module and the traceroute measurements data model. There are very few exceptions to this which are listed below. Also naming of information elements is identical between both models with a few exceptions. For the traceroute measurements data model, a few information elements have been added, some because of the different structure and some to provide additional information on completed measurements.

C.1. Naming

Basically, names in both models are chosen using the same naming conventions.

For the traceroute measurement configuration information all names, such as CtlProbesPerHop, are identical in both models except for the traceRoute prefix that was removed to avoid unnecessary redundancy in the XML file and for CtlDataSize which was renamed to CtlProbeDataSize for clarification in the traceroute measurements data model.

Results of measurements in the DISMAN-TRACEROUTE-MIB modules are distributed over two tables, the traceRouteResultsTable containing mainly information about ongoing measurements and the traceRouteProbeHistoryTable containing only information about completed measurements. According to the SMIv2 naming conventions names of information elements in these tables have different prefixes (traceRouteResults and traceRouteProbeHistory). Since the traceroute measurements data model only reports on completed measurements, this separation is not needed anymore and the prefix "Results" is used for all related information elements.

Beyond that, there are only a few changes in element names. The renaming actions include:

- traceRouteProbeHistoryProbeIndex to IndexPerHop,
- traceRouteProbeHistoryResponse to RoundTripTime,
- traceRouteProbeHistoryTime to ResultsEndDateAndTime,
- traceRouteProbeHistoryLastRC to ResultsHopRawOutputData.

C.2. Semantics

The semantics was changed for two information elements only.

For traceRouteProbeHistoryResponse in the DISMAN-TRACEROUTE-MIB, a value of 0 indicated, that it was not possible to transmit a probe. For the traceroute measurements data model, a value of 0 for element RoundTripTime indicates that the measured time was less than one millisecond, while for the case that it was not possible to transmit a probe a string is used that indicates the problem.

For traceRouteCtlIfIndex in the DISMAN-TRACEROUTE-MIB, a value of 0 indicated, that it the option to set the index is not available. This was translated to the traceroute measurements data model, such that a value of 0 for this element indicates that the used interface is unknown.

The element traceRouteProbeHistoryLastRC in the DISMAN-TRACEROUTE-MIB was replaced by element ResultsHopRawOutputData. While traceRouteProbeHistoryLastRC just reports a reply code, ResultsHopRawOutputData reports the full raw output data (per hop) produced by the traceroute measurements that was used.

C.3. Additional Information Elements

Only a few information elements have been added to the model of the DISMAN-TRACEROUTE-MIB module.

- For providing geographical information about hops in the traceroute measurement path, HopGeoLocation was added.
- For providing information on the MPLS label stack entries of a probe in the traceroute measurement path MPLSLabelStackEntry was added.
- For providing additional timestamp beyond ResultsEndDateAndTime, ResultsStartDateAndTime and Time were added.

Authors' Addresses

Saverio Niccolini
NEC Laboratories Europe, NEC Europe Ltd.
Kurfuersten-Anlage 36
Heidelberg 69115
Germany

Phone: +49 (0) 6221 4342 118
Email: saverio.niccolini@netlab.nec.de
URI: http://www.netlab.nec.de

Sandra Tartarelli
NEC Laboratories Europe, NEC Europe Ltd.
Kurfuersten-Anlage 36
Heidelberg 69115
Germany

Phone: +49 (0) 6221 4342 132
Email: sandra.tartarelli@netlab.nec.de
URI: http://www.netlab.nec.de

Juergen Quittek
NEC Laboratories Europe, NEC Europe Ltd.
Kurfuersten-Anlage 36
Heidelberg 69115
Germany

Phone: +49 (0) 6221 4342 115
Email: quittek@netlab.nec.de
URI: http://www.netlab.nec.de

Martin Swany
Dept. of Computer and Information Sciences, University of Delaware
Newark DE 19716
U.S.A.

Email: swany@UDel.Edu

Full Copyright Statement

Intellectual Property

http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).