TRISUL NETWORK ANALYTICS

http://trisul.org

TRP (Trisul Remote Protocol) Reference Documentation

version 4.0

Table of Contents

1 Introduction

Trisul Remote Protocol

2 Messages overview

A TRP message corresponds to a specific type of data you want to retrieve from Trisul. Every message sent will elicit a corresponding response or an ErrorResponse.

Visit Trisul Remote Protocol development guide home

2.1 Protocol messages

Hello	Get identity of Trisul server
CounterGroupInfo	Configuration info about counter groups
CounterItem	Traffic statistics of items keys
CounterGroup	Toppers in a counter group
SearchKeys	Search for items
FilteredDatagrams	Get raw packets matching a certain filter
SessionItem	Get details about a single flow
SessionGroup	Get top flows
SessionTracker	Retrieve flows for a session tracker
UpdateKeys	Set a friendly name for a key (eg 192.168.1.1 is GATEWAY_1
AlertItem	Details about a single alert (IDS, TCA, Flow, Malware)
AlertGroup	Search for alerts based on various criteria
ResourceItem	Details about a single resource item
ResourceGroup	Search for resources in a group (eg, HTTP URI, DNS)
KeyLookup	Lookup a key based on a name (eg a hostname)
KeySpace	Search for key activity within a range
Grep	Search for a text or binary pattern in reassembled TCP streams
ControlledContext	Perform a drilldown analysis of past traffic by controlling for an item. Ex : Analysis only of ICMP traffic in past 24 hours
ServerStats	CPU, Memory,Packet drops,and other system statistics of the Trisul server

2.2 Common messages

These are supporting messages the appear in one or more responses.

OKResponse	Common response type to certain request messages
ErrorResponse	Displays error response for all request type in case of error
KeyDetails	Returns a label/description for the item
KeyStats	Returns the keyitem values (contextID,countergroupID,...)
Timestamp	Specify a time instant for the request
TimeInterval	Specify a start time and end time for the request (used for certain request types)
CounterGroupDetails	Displays the details for the requested counter group
StatsTuple	A single statistic sample a timestamp + sample value tuple
MeterValues	Statistics for a set of meters
SessionID	Uniquely identifies a session (flow)

3 Hello

A handshake to get the identity of the Trisul server and to set the identity of the TRP client. It is not required that you start off a TRP Session with a HelloRequest, use this only to get server identity and set your own identity in server logs.

3.1 HelloRequest

```
message HelloRequest{
  required string   station_id=1;
}
```

station_id	A string representing the client application

3.2 HelloResponse

```
message HelloResponse{
  required string       trisul_id=1;
  required string       trisul_description=2;
  required string       connection_id=3;
  required string       version_string=4;
  required Timestamp    connection_start_time=5;
  required Timestamp    connection_up_time=6;
  required AuthLevel    current_auth_level=7;
}
```

Field	Description
trisul_id	The identity of the Trisul instance as found in the ProbeID parameter in the trisulConfig.xml file
trisul_description	A longer description found in the ProbeDesc parameter in trisulConfig.xml
connection_id	An ID generated by Trisul representing this TRP Session. The log files will contain this ID as a way to track activity
version_string	The version number of Trisul
connection_start_tim	When did this TRP session start

Field	Description
e	
connection_up_time	How long has thisTRP session been running
current_auth_level	Current authentication level (admin/forensics/basic) based on the certificate produced by the client

4 CounterGroupInfo

Use this to retrieve information about Counter Groups such as bucket sizes, time window of available data, names, and guid. The most common usage of this message is to retrieve the bucket size and the available time window.

4.1 CounterGroupInfoRequest

```
message CounterGroupInfoRequest {
    optional int64         context=1[default=0];
    optional string        counter_group=2;
}
```

Usage

- Skip the counter_group field if you want to retrieve details for all counter groups.

context	The context
counter_group	The counter group id (a guid),skip this to retrieve information about all counter groups

4.2 CounterGroupInfoResponse

```
message CounterGroupInfoResponse{
    optional int64                  context=1;
    repeated CounterGroupDetails    group_details=2;
}
```

context	The context
group_details	Details of requested counter groups in a CounterGroupDetails message

Samples

See cginfo on the Trisul Scripts github page.

5 CounterItem

Retrieve traffic statistics about a counter item. A counter item is identified by a counter group id (a GUID) and a key. The typical use of this message is to get time-series usage or total usage of any metered item.

5.1 CounterItemRequest

```
message CounterItemRequest{
    optional int64          context=1 [default=0];
    required string         counter_group=2;
    optional int64          meter=3;
    required string         key=4;
    required TimeInterval   time_interval=5;
    optional int64          volumes_only=6 [default=0];
}
```

context	Context ID
counter_group	A counter group guid
meter	A meter id To see list of available meters for each counter group, go to Customize > Counters > Click on Edit Topper Policies. Omit this field if you want to retrieve data for all meters
key	A key that identifies the counter item
time_interval	The desired time interval
volumes_only	Set this to 1 if you only want to report Totals for each meter. Use this to retrieve total volume of data in the entire time interval

5.2 CounterItemResponse

```
message CounterItemResponse{
    required KeyStats       stats=1;
}
```

stats	The requested traffic statistics in a KeyStats message

Sample code

See getvolume and hourlystats on the Trisul Scripts github page.

6 CounterGroup

Retrieve toppers for a specific meter in counter group in a specified time interval.

Some examples :

1. Retrieve top hosts by total volume yesterday
2. Retrieve top applications by connections between 8AM and 6PM today

6.1 CounterGroupRequest

```
message CounterGroupRequest{
    optional int64          context=1 [default=0];
    required string         counter_group=2;
    optional int64          meter=3 [default=0];
    optional int64          maxitems=4 [default=10];
    optional TimeInterval   time_interval=5;
    optional Timestamp      time_instant=6;
    optional int64          flags=7;
}
```

Field	Description
context	Context ID
counter_group	The counter group id
meter	The meter or StatID for which toppers are tracked
maxitems	Number of toppers to return
time_interval	The time interval in which you want to compute the toppers
flags	Reserved

6.2 CounterGroupResponse

Returns the list of keys that represent the top items. If you want to retrieve statistics for these

keys you can send out individual CounterItemRequests for each of them. The field metric contains the total volume for each key.

```
message  CounterGroupResponse {
    required int64          context=1;
    required string         counter_group=2;
    required int64          meter=3;
    repeated KeyDetails     keys=6;
}
```

Field	Description
context	content ID
counter_group	The counter group ID
meter	The requested meter
keys	A list of KeyDetails that represent the top keys. Within the *KeyDetails* message the field metric contains the aggrgate value. For rate counters, you need to multiply that value by the BucketSize to get the total bytes

7 BulkCounterItem

Similar to CounterItem but used to get a number of keys from the same countergroup, time interval and meter at once.

7.1 BulkCounterItemRequest

Get traffic stats of multiple counter items at once. Use this method if you want to retrieve stats for a number of items for the same time interval and meter.

```
message BulkCounterItemRequest{
    optional int64          context=1[default=0];
    required string         counter_group=2;
    required int64          meter=3;
    required TimeInterval   time_interval=4;
    repeated string         keys=5;
}
```

7.2 BulkCounterItemResponse

```
message BulkCounterItemResponse{
    repeated KeyStats       stats=1;
}
```

Field	Description
stats	An array of KeyStats messages

8 SearchKeys

Find keys matching a given pattern. You can use this to retrieve all hosts which have a youtube in their DNS names for example.

8.1 SearchkeysRequest

```
message SearchKeysRequest{
    optional int64          context=1[default=0];
    required string         counter_group=2;
    required string         pattern=3;
    required int64          maxitems=4;
}
```

context	Context ID
counter_group	The counter group ID
pattern	The search string (dont use regex expressions)
maxitems	Number of matches to retrieve

8.2 SearchKeysResponse

```
message SearchKeysResponse{
    optional int64          context=1;
    required string         counter_group=2;
    repeated KeyDetails     found_keys=3;
}
```

context	Context ID
counter_group	The counter group ID
found_keys	A list of matching keys

9 FilteredDatagram

Retrieve raw packets from Trisul. You can retrieve them in libpcap format ready to be opened in Wireshark or Unsniff.

Usage

The FilteredDatagram message allows you to :

1. Retrieve packets for a given flow (TCP/UDP session)
2. Retrieve relevant packets for an alert
3. Retrieve packets for a resource (such as a HTTP URI)
4. Arbitrary filter string
5. Download the packets inLIBPCAP format
6. Save the PCAP on the server if the result set is expected to be huge (>1GB)

The filter string

Allows you to use a powerful filter to get the packets you want. The filter string in an expression in Trisul Filter Format. You can retrieve packets matching exotic metering criteria like â€œGet me all non-HTTP traffic from China or Ukrain

9.1 FilteredDatagramRequest

```
message FilteredDatagramRequest{
    optional int64          max_packets=1[default=0];
    optional int64          max_bytes=2[default=0];
    optional CompressionType compress_type=3[default=UNCOMPRESSED];
     // by trisul filter format expr
    message ByFilterExpr {
        required TimeInterval   time_interval=1;
        required string         filter_expression=2;
    }
    optional ByFilterExpr   filter_expression=4;
   // by session
    message BySession   {
```

```
      optional string         session_group=1[default="{99A787..}"]; // IP Flows
      required SessionID      session_id=2;
    }
    optional BySession          session=5;
     // by alert
    message ByAlert     {
      optional string         alert_group=1[default="{9AFD8C08-...}"]; // IDS
      required AlertID        alert_id=2;
    }
    optional ByAlert            alert=6;

    // by resource
    message ByResource  {
      required string         resource_group=1;
      required ResourceID     resource_id=2;
    }
    optional ByResource         resource=7;

    optional PcapDisposition    disposition=8[default=DOWNLOAD];
}
```

max_packets	Maximum number of packets to retrieve
max_bytes	Maximum number of bytes to retrieve
compress_type	NONE = dont compress, GZIP = compress using gzip
time_interval	Time interval of interest a TimeInterval object
filter_expression	An expression in Trisul Filter Format
disposition	DOWNLOAD (default) = download to client, SAVE_ON_SERVER = leave on server

Usage notes

- If both max_packets and max_bytes are specified, Trisul will stop at whatever limit is reached first

9.2 FilteredDatagramResponse

```
message FilteredDatagramResponse{
    required PcapFormat         format=1;
    required CompressionType    compress_type=2;
    required TimeInterval       time_interval=3;
    required int64              num_datagrams=4;
    required int64              num_bytes=5;
    required string             sha1=6;
    required bytes              contents=7;
    required PcapDisposition    disposition=8;
    optional string             path=9;


}
```

format	LIBPCAP = the contents string is a file in libpcap format
compress_type	Type of compression employed on the contents string
time_interval	Time interval retrieved
num_datagrams	Packets retrieved
num_bytes	Bytes retrieved
sha1	
contents	The requested packet data
disposition	Whatever was requested, DOWNLOAD or SAVE_ON_SERVER
path	If disposition=SAVE_ON_SERVER, Where is the PCAPsaved ?

The contents field is a string that contains the packet data.

- If compression type is GZIP, you need to unzip the contents string
- Just save the uncompressed contents string into a file
- You can then open the file using Wireshark or Unsniff

10 SessionItem

Retrieve information about a single flow.

10.1 SessionItemRequest

```
message SessionItemRequest{
    optional int64          context=1[default=0];
    optional string         session_group=2[default="{99A78737-4B41-4387-8F31-
8077DB917336}"];
    repeated string         session_keys=3;
    repeated SessionID      session_ids=4;
}
```

Each flow is uniquely identified by a

1. Session ID : For completed or long running flows (OR)
2. Session Key : For brand new sessions (less than 1 minute old) in progress

context	Context ID
session_group	
session_keys	The session key string if available
session_ids	The session key

Note either the session_key or session_id must be specified.

10.1.1 SessionItemResponse

```
message SessionItemResponse{
    optional int64          context=1[default=0];
    required string         session_group=2;
    message Item {
        optional string         session_key=1;
        optional SessionID      session_id=2;
        optional string         user_label=3;
        required TimeInterval   time_interval=4;
        required int64          state=5;
        required int64          az_bytes=6;
        required int64          za_bytes=7;
        required KeyDetails     key1A=8;
        required KeyDetails     key2A=9;
        required KeyDetails     key1Z=10;
        required KeyDetails     key2Z=11;
    }
    repeated Item           items=3;
}
```

context	context ID
session_group	GUID of session group
session_key	The key if the flow is very recent and does not have a persistent ID assigned yet
session_id	The persistent ID of the flow (consists of a Slice number & Session ID
user_label	Flow label if available
time_interval	Flow time
state	Flow state
az_bytes	Number of bytes transferred from A-End of the flow to Z-End of the flow
za_bytes	Bytes from Z-End to A-End

Key1A	A-end key details (IP Address)
key2A	A-end key details (Port)
key1Z	Z-end IP
Key2Z	Z-end Port

11 SessionGroup

Retrieve current top flows.

11.1 SessionGroupRequest

```
message SessionGroupRequest {
    optional int64          context=1;
    optional string         session_group=2;
    optional int64          tracker_id=3;
    optional string         key_filter=4;
    optional int64          maxitems=5 [default=100];
}
```

context	context ID
session_group	GUID of session group , use *{99A78737-4B41-4387-8F31-8077DB917336}* for TCP flows
tracker_id	If present, only flows matching a tracker are returned
key_filter	If present, only flows which have an endpoint with this key are returned. For example, if this field is HTTP (p-0050), then only the top HTTP flows are returned
maxitems	Number of matches to return

11.2 SessionGroupResponse

```
message SessionGroupResponse {
    optional int64          context=1;
    required string         session_group=2;
    repeated string         session_keys=3;
}
```

context	context ID
session_group	GUID of session group , use *{99A78737-4B41-4387-8F31-8077DB917336}* for TCP flows
session_keys	List of session keys. Note that we do not return SessionIDs because we are dealing with new flows which may not have a persistent session id assigned by Trisul yet

Typically, after getting the SessionIDs you will fire individual SessionItemRequests for each SessionKey in order to retrieve the flow details.

12 SessionTracker

Retrieve session trackers. Session trackers are tools which enable Trisul to automatically track and store top flows matching a given criteria.

12.1 SessionTrackerRequest

```
message SessionTrackerRequest {
    optional int64          context=1;
    optional string         session_group=2;
    required int64          tracker_id=3 [default=1];
    optional int64          maxitems=4 [default=100];
    required TimeInterval   time_interval=5;
}
```

context	context ID
session_group	GUID of session group , use *{99A78737-4B41-4387-8F31-8077DB917336}* for TCP flows
tracker_id	The session tracker id
maxitems	Number of items to return
time_interval	The time window

12.2 SessionTrackerResponse

```
message SessionTrackerResponse{
    optional int64          context=1;
    required string         session_group=2;
    repeated SessionID      sessions=3;
}
```

context	context ID
session_group	The GUID of the session group
sessions	The list of sessions

Typically, after getting the SessionIDs you will fire individual SessionItemRequests for each item in order to retrieve the details.

13 UpdateKeys

Used to assign a friendly name and/or description to a key.

13.1 UpdateKeysRequest

```
message UpdateKeyRequest{
    optional int64          context=1;
    required string         counter_group=2;
    required string         key=4;
    required string         label=5;
    optional string         description=6;
}
```

context	context ID
counter_group	The counter group ID
key	The key
label	The label you wish to assign
description	Description, if any

13.2 UpdateKeysResponse

The response to this message is

1. OKResponse : If the update was successful
2. ErrorResponse : If an error occurred

14 AlertItem

Used to retrieve details about a list of alert ids.

14.1 AlertItemRequest

Use this request to get details about a list of alerts_ids.

```
message AlertItemRequest{
    optional int64          context=1[default=0];
    required string         alert_group=2;
    repeated AlertID        alert_ids=3;
}
```

context	context ID
alert_group	The GUID of the alert group
alert_ids	The list of alert_ids

14.2 AlertItemResponse

Details about each alert in the request. Invalid IDs in the request will not be present in the response.

```
message AlertItemResponse{
    optional int64          context=1;
    required string         alert_group=2;
    message Item {
        optional int64          sensor_id=1;
        required Timestamp      time=2;
        required AlertID        alert_id=3;
        optional string         source_ip=4;
        optional string         source_port=5;
        optional string         destination_ip=6;
        optional string         destination_port=7;
        required string         sigid=8;
        required string         classification=9;
        required string         priority=10;
        required Timestamp      dispatch_time=11;
        required string         aux_message1=12;
        required string         aux_message2=13;
    }
    repeated Item           items=3;
}
```

context	CONTEXT ID
alert_group	The GUID of Alert group
sensor_id	
time	Time instant retrieved
source_ip	
source_port	
destination_ip	

destination_port	
sigid	
classification	
priority	
dispatch_time	
aux_message1	
aux_message2	
items	

15 AlertGroup

The AlertGroup Request and Response methods are used to query the alerts known to Trisul.

Trisul supports four different types of alerts

1. Threshold Crossing Alerts
2. Flow Tracker Alerts
3. Malware Blacklist Alerts (requires the Badfellas plugin)
4. Snort or Suricata Alerts (Trisul accepts these alerts from Unix socket)

Each of these categories are called Alert Groups and each of them is uniquely identified by a GUID. You need to specify an appropriateGUID for the *alert_group* parameter, see here for the GUID definitions.

15.1 AlertGroupQueryRequest

Retrieve alerts matching a query.

1. If you specify one or more of the optional parameters they will be ANDed.
2. If you dont specify any parameter all alerts will be retrieved

Note that all the parameters must be in Trisul Key Format. Eg use C0.A8.01.01 instead of 192.168.1.1. See code samples for automatically translating human readable values into these keys

```
message AlertGroupRequest {
    optional int64          context=1[default=0];
    required string         alert_group=2;
    required TimeInterval   time_interval=3;
    optional int64          maxitems=5 [default=10];
    optional string         source_ip=6;
    optional string         source_port=7;
    optional string         destination_ip=8;
    optional string         destination_port=9;
    optional string         sigid=10;
    optional string         classification=11;
```

```
    optional string         priority=12;
    optional string         aux_message1=13;
    optional string         aux_message2=14;
}
```

context	
alert_group	
time_interval	
maxitems	
source_ip	
source_port	
destination_ip	
destination_port	
sigid	If alert_group = IDS alerts this is usually s-snortid. Eg : *s-16801*
classification	
priority	
aux_message1	Contains detailed information about the Threshold Crossing / Flow Tracker/ Malware alert
aux_message2	

15.2 AlertGroupResponse

Matching alert IDs. You will usually follow this up with an //AlertItemRequest// to get full details about each alert in the list.

```
message AlertGroupResponse {
    optional int64          context=1;
    required string         alert_group=2;
    repeated AlertID        alerts=3;
}
```

context	
alert_group	
alerts	

16 ResourceItem

Resource Items represent a single resource (such as HTTP URL or DNS name).

Use this request to retrieve details of each resource based on the ID.

16.1 ResourceItemRequest

```
message ResourceItemRequest{
    optional int64          context=1[default=0];
    required string         resource_group=2;
    repeated ResourceID     resource_ids=3;
}
```

context	
resource_group	
resource_ids	A list of resource ids Usually from an earlier ResourceGroupRequest query for matching resources

16.2 ResourceItemResponse

```
message ResourceItemResponse{
    optional int64          context=1;
    required string         resource_group=2;
    message Item {
        required Timestamp      time=1;
        required ResourceID     resource_id=2;
        optional string         source_ip=3;
        optional string         source_port=4;
        optional string         destination_ip=5;
        optional string         destination_port=6;
        optional string         uri=7;
        optional string         userlabel=8;
    }
    repeated Item           items=3;
}
```

Field	Description
context	
resource_group	
time	
resource_id	
source_ip	
source_port	
destination_ip	
destination_port	
uri	
userlabel	

17 ResourceGroup

Resources are additional meta data collected by Trisul about the traffic it sees. Currently Trisul logs the following resources

1. DNS names
2. HTTP URLs
3. SSL Certificates

You use the ResourceGroupRequest to find the resources you need- followed by one or more ResourceItemRequests to retrieve details of those resources.

17.1 ResourceGroupRequest

```
message ResourceGroupRequest {
    optional int64          context=1[default=0];
    required string         resource_group=2;
    required TimeInterval   time_interval=3;
    optional int64          maxitems=4 [default=10];
    optional string         source_ip=5;
    optional string         source_port=6;
    optional string         destination_ip=7;
    optional string         destination_port=8;
    optional string         uri_pattern=9;
    optional string         userlabel_pattern=10;
}
```

Field	Description
context	
resource_group	Use the Resource Group GUIDs or the constants `TrisulRP::Guids::RG_URL` or `TrisulRP::Guids::RG_DNS` .. etc
time_interval	A TimeInterval in which to search for matches
maxitems	
source_ip	

Field	Description
source_port	
destination_ip	
destination_port	
uri_pattern	A part of the resource name. Example : Part of URL or domain name.
userlabel_pattern	

17.2 ResourceGroupResponse

Retrieves a list of ResourceIDs matching the request query.

Typically after this response you need to issue one or more ResourceItemRequest requests to query the attributes of each resource.

```
message ResourceGroupResponse {
    optional int64          context=1;
    required string         resource_group=2;
    repeated ResourceID     resources=3;
}
```

context	
resource_group	
resources	A list of resource IDs

18 KeyLookup

Retrieve user labels for keys. Typical uses would be to get hostnames, alert signature names, app names, and any other names that apply to keys within a counter group.

18.1 KeyLookupRequest

```
message KeyLookupRequest {
    optional int64          context=1[default=0];
    required string         counter_group=2;
    repeated string         keys=3;
}
```

Field	Description
context	
counter_group	The counter group GUID to which these keys belong
keys	List of keys you want to resolve to names

18.2 KeyLookupResponse

```
message KeyLookupResponse {
    optional int64          context=1;
    required string         counter_group=2;
    repeated KeyDetails     key_details=3;
}
```

Field	Description
context	

Field	Description
counter_group	The counter group GUID to which these keys belong
key_details	List of KeyDetails containing labels for keys

19 KeySpace

These methods allow you to search for key activity in ranges. For example, helps you locate IP address in arbitrary IP blocks.

19.1 KeySpaceRequest

```
////////////////////////////////////
// KeySpaceRequest
message KeySpaceRequest {
    optional int64          context=1[default=0];
    required string         counter_group=2;
    required TimeInterval   time_interval=3;
    optional int64          maxitems=4 [default=100];
    message KeySpace {
        required string     from=1;
        required string     to=2;
    }
    repeated KeySpace       spaces=5;
}
```

Field	Description
context	
counter_group	The counter group GUID to which these keys belong
time_interval	
spaces	List of KeySpace messages, each containing a range of keys

19.2 KeySpaceResponse

```
message KeySpaceResponse  {
    optional int64          context=1;
    optional string         counter_group=2;
    repeated string         hits=3;
}
```

Field	Description
context	
counter_group	
hits	List of keys that matched

20 Grep

Search for text or binary patterns in reassembled TCP streams.

20.1 GrepRequest

```
message GrepRequest {
    optional int64          context=1[default=0];
    optional string         session_group=2[default="{99A78.."];
    required TimeInterval   time_interval=3;
    optional int64          maxitems=4 [default=50];
    required string         pattern=5;
}
```

context	always 0
session_group	always {99A78737-4B41-4387-8F31-8077DB917336} represents TCP
time_interval	Timeframe to search a TimeInterval object
maxitems	Number of items to match
pattern	Text string or binary sequence of bytes

All TCP streams are reassembled and matched with the sequence of bytes specified in // pattern //. Note that pattern is not a Regex, it must be an exact matching string.

20.2 GrepResponse

```
message GrepResponse  {
    optional int64          context=1;
    optional string         session_group=2[default="{99A78737-4B41-4387-8F31-
8077DB917336}"];
    repeated SessionID      sessions=3;
}
```

context	Always 1
session_group	Always TCP
sessions	List of SessionIDs

Normally you would follow up a GrepResponse with a SessionItemRequest to get more information about the returned flows.

21 ServerStats

Statistics about the Trisul server.

21.1 ServerStatsRequest

```
message ServerStatsRequest{
     required int64         param=1;
}
```

param	Reserved

21.2 ServerStatsResponse

```
message ServerStatsResponse{
    required string         instance_name=1;
    required int64          connections=2;
    required int64          uptime_seconds=3;
    required double         cpu_usage_percent_trisul=4;
    required double         cpu_usage_percent_total=5;
    required double         mem_usage_trisul=6;
    required double         mem_usage_total=7;
    required double         mem_total=8;
    required int64          size_total=9;
    required double         drop_percent_cap=11;
    required double         drop_percent_trisul=12;
    required TimeInterval   time_interval=13;
}
```

instance_name	Name of trisul probe as defined in trisulConfig.xml
connections	Number of TRP connections (including this one)
uptime_seconds	How long has this instance of Trisul been running
cpu_usage_percent_trisul	CPU usage of Trisul
cpu_usage_percent_total	Total CPU usage of machine
mem_usage_trisul	Memory used by Trisul
mem_usage_total	Total memory used
mem_total	Total physical memory installed
size_total	Unused
drop_percent_cap	Percentage of packets dropped by the capture mechanism (libpcap or linux-rx-ring)
drop_percent_trisul	Percentage of packets dropped by Trisul
time_interval	Data availability window

22 OKResponse

You can expect OKResponse back when some requests do not have a specific response message.

```
message OKResponse{
   required int64       original_command=1;
   optional string      message=2;
}
```

original_command	The original command code (see enums in trp.proto) that generated this OK response
message	A text string with more info

23 ErrorResponse

Any request message can get back an error response.

```
message ErrorResponse{
   required int64       original_command=1;
   required int64       error_code=2;
   required string      error_message=3;
}
```

original_command	The original command code (see enums in trp.proto) that generated this OK response
error_code	A numeric error code
error_message	A text error message

24 KeyDetails

Represents a key.

```
message KeyDetails {
    required string         key=1;
    optional string         label=2;
    optional string         description=3;
    optional int64          metric=4;
}
```

Field	Description
key	Key (eg p-0050)
label	Friendly name of key (eg http)
description	Description, if available
metric	A optional metric to go along with the key. Its usage depends on the request message.

25 KeyStats

Key statistics across multiple meters.

```
message KeyStats {
    optional int64          context=1[default=0];
    required string         counter_group=2;
    required string         key=3;
    repeated MeterValues    meters=4;
}
```

Field	Description
context[default=0]	The context id
counter_group	The GUID of the counter group containing the key
key	The key
meters	Time series values of meters in a MeterValues message

26 Timestamp

A time instant.

```
message Timestamp{
  required int64            tv_sec=1;
  optional int64            tv_usec=2 [default=0];
}
```

Field	Description
tv_sec	Unix seconds time since Jan 1 1970 GMT
tv_usec [default=0]	Optional microseconds

27 TimeInterval

A time interval, two time instants.

```
message TimeInterval  {
    required Timestamp      from=1;
    required Timestamp      to=2;
}
```

from	Start time a Timestamp object
to	End time

27.1 Usage

The long way to create a TimeInterval object is

```
# from and to are two "Time" objects
tmint = TimeInterval.new(
    :from => Timestamp.new(:tv_sec => from.tv_sec),
    :to => Timestamp.new(:tv_sec => to.tv_sec)
)
```

The TrisulRP Ruby Gem has a shortcut called mk_time_interval

```
tmint = mk_time_interval( [from, to] )
```

28 CounterGroupDetails

Details about a counter group.

```
message CounterGroupDetails {
    required string         guid=1;
    required string         name=2;
    optional int64          bucket_size=3;
    optional TimeInterval   time_interval=4;
    optional int64          topper_bucket_size=5;
}
```

Field	Description
guid	The GUID which uniquely identifies this counter group
name	Name
bucket_size	Statistics bucket size seconds
time_interval	TimeInterval for which data is available for this counter group
topper_bucket_size	Topper bucket size seconds

29 StatsTuple

A timestamp + sample value tuple.

```
message StatsTuple {
    required Timestamp      ts=1;
    required int64          val=2;
}
```

Field	Description
ts	Time instant, t a TimeStamp message
val	Value at t

30 MeterValues

A time-series data for a single meter. A meter is a statistical data point such as Total Bytes, Received,Total Connections etc. Each counter group hosts a number of such meters. You can get a list of these meters from the web interface via (Customize -> Counters -> Edit Topper Policies)

```
message MeterValues {
    required int32          meter=1;
    repeated StatsTuple     values=2;
}
```

Field	Description
meter	The meter ID (or stat ID)
values	A list of StatsTuples (timestamp + sample value)

31 SessionID

Uniquely identifies a session over all time.

```
message SessionID {
    required int64          slice_id=1;
    required int64          session_id=2;
}
```

Field	Description
slice_id	A database slice id
session_id	A session id

32 Resource ID

Uniquely identifies a resource over all time.

```
message ResourceID {
    required int64          slice_id=1;
    required int64          resource_id=2;
}
```