Enrich PMACCTD netflow data with KAFKA/KSQL

##Software used pmacctd compiled with

```
NetFlow Accounting Daemon, nfacctd 1.7.8-git [20211210-0 (db67590d)]

Arguments:
 '--prefix=/opt/pmacct' '--enable-jansson' '--enable-kafka' '--enable-avro' '--
enable-serdes' 'AVRO_CFLAGS=-I/usr/local/include/avro' 'AVRO_LIBS=-
L/usr/local/lib64 -lavro' '--enable-l2' '--enable-traffic-bins' '--enable-bgp-
bins' '--enable-bmp-bins' '--enable-st-bins'
```

confluent-ksqldb confluent-ksql confluent-kafka confluent-schema-registry

nfacctd:

```
cat netflow.conf | grep -v '#'
daemonize: false
debug: false
nfacctd_port: 4739
nfacctd_disable_checks: true
nfacctd_renormalize: true
nfacctd_time_new: true
nfacctd_net: bgp
nfacctd_as_new: bgp
sampling_map: /opt/pmacct/etc/v0.1/sampling.map
bgp_daemon: true
bgp_daemon_ip: 10.208.2.27
bgp_daemon_max_peers: 100
bgp_aspath_radius: 5
bgp_peer_src_as_type: bgp
bgp_src_as_path_type: bgp
bgp_src_std_comm_type: bgp
bgp_src_ext_comm_type: bgp
bgp_src_as_path_type: bgp
bgp_src_local_pref_type: bgp
bgp_src_med_type: bgp
bgp_follow_default: true
bgp_daemon_msglog_file: /tmp/bgp-peer.log
plugins: kafka[data]
aggregate[data]:src_host,dst_host,src_mask,dst_mask,src_as,dst_as,tos,proto,in_ifa
ce,out_iface,std_comm,ext_comm,local_pref,med,src_std_comm,src_ext_comm,src_local_
pref,src_med,mpls_vpn_rd,mpls_label_top,mpls_label_bottom,mpls_stack_depth,timesta
mp_start,timestamp_end,timestamp_arrival,timestamp_export,vrf_id_ingress,vrf_id_eg
ress,mtla,src_port,dst_port,tcpflags,forwarding_status,vrf_name,label,peer_dst_ip
kafka_output[data]: avro
kafka_topic[data]: data_host_raw_1m
kafka_refresh_time[data]: 60
```

```
kafka_history[data]: 1m
kafka_history_roundoff[data]: m
kafka_avro_schema_registry[data]: http://kafka-uk:8081
kafka_broker_host[data]: kafka-uk
aggregate_primitives: /opt/pmacct/etc/v0.1/primitives.lst
flow_to_rd_map: /opt/pmacct/etc/v0.1/flow_to_rd.map
pre_tag_map: /opt/pmacct/etc/v0.1/pretag.map
maps_entries: 10000
```

> ***NOTE:*** /opt/pmacct/etc/v0.1/primitives.lst is for not yet supported primitives in nfacctd

```
cat primitives.lst
name=vrf_id_ingress     field_type=234  len=4   semantics=u_int
name=vrf_id_egress      field_type=235  len=4   semantics=u_int
name=mtla      field_type=47  len=4   semantics=ip
name=forwarding_status field_type=89 len=1 semantics=u_int
name=vrf_name field_type=236 len=32 semantics=str
```

> ***NOTE:*** Pretag is used here to map the netflow src ip of the exporter (router) to a name

```
cat pretag.map | grep lab-erta-ndc
set_label=lab-erta-ndc ip=10.192.180.1
```

> ***NOTE:*** For nfacctd to be able to do a rib lookup for VPNv4 , we need to know the vrf RD. Below is used where for each router and mpls_vrn_id we put the ID = RD (type 0 or type 1:RD)

```
cat flow_to_rd.map | grep 10.192.180.1
id=0:1:500              ip=10.192.180.1         mpls_vpn_id=3
id=0:1:500              ip=10.192.180.1         mpls_vpn_id=4
id=0:1:600              ip=10.192.180.1         mpls_vpn_id=5
id=0:800:500            ip=10.192.180.17                mpls_vpn_id=3
id=0:800:500            ip=10.192.180.17                mpls_vpn_id=5
id=0:800:600            ip=10.192.180.17                mpls_vpn_id=4
id=0:800:500            ip=10.192.180.18                mpls_vpn_id=4
id=0:800:500            ip=10.192.180.18                mpls_vpn_id=5
id=0:800:600            ip=10.192.180.18                mpls_vpn_id=6
```

Resulting entries in kafka

```
rowtime: 2021/12/12 11:54:01.103 Z, key: <null>, value:
{
  "label": "lab-erta-ndc",
  "as_src": 0,
  "as_dst": 0,
  "comms": "",
```

```
    "ecomms": "RT:800:500",
    "local_pref": 100,
    "med": 0,
    "peer_ip_dst": "10.192.180.1",
    "comms_src": "",
    "ecomms_src": "RT:800:500",
    "local_pref_src": 100,
    "med_src": 0,
    "iface_in": 23,
    "iface_out": 0,
    "mpls_vpn_rd": "0:1:500",
    "ip_src": "10.195.199.3",
    "ip_dst": "10.195.199.11",
    "mask_src": 32,
    "mask_dst": 32,
    "port_src": 179,
    "port_dst": 50280,
    "tcp_flags": "0",
    "ip_proto": "tcp",
    "tos": 192,
    "mpls_label_top": 0,
    "mpls_label_bottom": 0,
    "mpls_stack_depth": 0,
    "timestamp_start": "2021-12-12 11:53:27.370000",
    "timestamp_end": "2021-12-12 11:53:27.380000",
    "timestamp_arrival": "2021-12-12 11:53:43.041531",
    "timestamp_export": "2021-12-12 11:53:43.000000",
    "custom_primitives": {
      "forwarding_status": "64",
      "vrf_name": "",
      "vrf_id_ingress": "4",
      "vrf_id_egress": "0",
      "mtla": "0.0.0.0"
    },
    "stamp_inserted": "2021-12-12 11:53:00",
    "stamp_updated": "2021-12-12 11:54:01",
    "packets": 2000,
    "flows": null,
    "bytes": 123000,
    "writer_id": "data/748390"
  }
```


Now that flowdata is in kafka time to enrich 3 fields. iface_in with iface_in_name iface_out with iface_name and forwarding status with forwarding_status_name.

A mapping between ifIndex and Ifname per device is needed for first two.

A mapping between forwarding status decimal value and forwarding status as per RFC7220

A simple python script is used to take the ifIndex,ifName,router from either SOT or NMS and write to a kafka topic "databus_interfaces".

Same python will write the forwarding status mapping to kafka topic "forwarding_status".

Enrich helper topics are now in kafka.

Time to enrich. In terms of stream enrichment there are many open source options ranging from kafka-stream, apache flink , apache spark to confluent KSQL. KSQL abstracts the java code and provides a SQL like interface to create streams , tables and allows joins stream-to-stream (windowed) or stream to tables (non windowed).

So let's begin.

1. Take nfacctd topic and create a stream

```
ksql> SET 'auto.offset.reset' = 'earliest';
ksql> create stream STREAM_DATA_HOST_RAW_1M
WITH (kafka_topic='data_host_raw_1m', partitions=1, value_format='AVRO');
```

You need to add the avro schema for the topic. This is as easy as copying the current one that pmacctd writes to your topic.

```
curl  --noproxy '*' -X POST -H "Content-Type:
application/vnd.schemaregistry.v1+json" \
--data "{\"schema\": $(curl  --noproxy '*' -s http://kafka-
uk:8081/subjects/data_host_raw_1m-acct-data-value/versions/latest | jq
'.schema')}" \
http://kafka-uk:8081/subjects/data_host_raw_1m-value/versions
```

- Verification

```
ksql> select label,iface_in,iface_out from STREAM_DATA_HOST_RAW_1M emit changes
limit 10;
+-----------------------------------------+-----------------------------------
--------+-----------------------------------------+
|LABEL                                    |IFACE_IN
|IFACE_OUT                                |
+-----------------------------------------+-----------------------------------
--------+-----------------------------------------+
|lab-erta-ndc                             |101
|10                                       |
|lab-erta-ndc                             |10
|5                                        |
|lab-erta-ndc                             |10
|7                                        |
|lab-erta-ndc                             |10
|7                                        |
|lab-erta-ndc                             |10
|5                                        |
|lab-erta-ndc                             |10
|7                                        |
```

```
|lab-erta-ndc                                |10
|7                                           |
|lab-erta-ndc                                |10
|0                                           |
|lab-erta-ndc                                |10
|0                                           |
Limit Reached
Query terminated
```

2. Transform the stream with two fields which will be used as keys to join later on.

```
#create stream for rekeyed
CREATE STREAM STREAM_DATA_HOST_RAW_1M_REKEY with (VALUE_FORMAT='JSON',
KAFKA_TOPIC='STREAM_DATA_HOST_RAW_1M_REKEY')
AS SELECT
    CAST(STREAM_DATA_HOST_RAW_1M.iface_in AS VARCHAR)
        + '_' + STREAM_DATA_HOST_RAW_1M.label AS iface_in_key,
    CAST(STREAM_DATA_HOST_RAW_1M.iface_out AS VARCHAR)
        + '_' + STREAM_DATA_HOST_RAW_1M.label AS iface_out_key ,
        *
from STREAM_DATA_HOST_RAW_1M
EMIT CHANGES;
```

2.1 Verification

```
ksql> select label,iface_in_key,iface_out_key from STREAM_DATA_HOST_RAW_1M_REKEY
emit changes limit 10;
+-------------------------------------------+-----------------------------------
--------+-------------------------------------------+
|LABEL                                      |IFACE_IN_KEY
|IFACE_OUT_KEY                              |
+-------------------------------------------+-----------------------------------
--------+-------------------------------------------+
|lab-erta-ndc                               |101_lab-erta-ndc
|10_lab-erta-ndc                            |
|lab-erta-ndc                               |10_lab-erta-ndc
|101_lab-erta-ndc                           |
|lab-erta-ndc                               |10_lab-erta-ndc
|5_lab-erta-ndc                             |
|lab-erta-ndc                               |10_lab-erta-ndc
|7_lab-erta-ndc                             |
|lab-erta-ndc                               |10_lab-erta-ndc
|7_lab-erta-ndc                             |
|lab-erta-ndc                               |10_lab-erta-ndc
|7_lab-erta-ndc                             |
|lab-erta-ndc                               |10_lab-erta-ndc
|7_lab-erta-ndc                             |
|lab-erta-ndc                               |5_lab-erta-ndc
|0_lab-erta-ndc                             |
```

```
|lab-erta-ndc                                    |10_lab-erta-ndc
|7_lab-erta-ndc                                  |
|lab-erta-ndc                                    |10_lab-erta-ndc
|5_lab-erta-ndc                                  |
Limit Reached
Query terminated
```

3. Create stream stream_forwarding_status from topic forwarding_status

```
CREATE stream stream_forwarding_status
(ID VARCHAR ,LABEL VARCHAR)
WITH (kafka_topic='forwarding_status', partitions=1, value_format='json');
```

3.1 Verification

```
ksql> select * from stream_forwarding_status EMIT CHANGES limit 3;
+-------------------------------------------------------------------------+------------
-----------------------------------------------------+
|ID                                                                       |LABEL
|
+-------------------------------------------------------------------------+------------
-----------------------------------------------------+
|64                                                                       |Forwarded
|
|128                                                                      |Dropped
|
|192                                                                      |Consumed
|
Limit Reached
Query terminated
```

3.2 rekey stream_forwarding_status to stream_forwarding_status_rekey by id

```
CREATE stream stream_forwarding_status_rekey AS
 SELECT * FROM stream_forwarding_status \
 PARTITION BY id;
```

3.3 create table table_forwarding_status

```
CREATE TABLE table_forwarding_status
(ID VARCHAR PRIMARY KEY ,LABEL VARCHAR)
WITH (
     KAFKA_TOPIC = 'STREAM_FORWARDING_STATUS_REKEY',
```

```
      VALUE_FORMAT = 'JSON'
    );
```

3.4 Verification

```
ksql> select * from table_forwarding_status EMIT CHANGES limit 3 ;
+-----------------------------------------------------------------------+------------
-------------------------------------------------------+
|ID                                                                     |LABEL
|
+-----------------------------------------------------------------------+------------
-------------------------------------------------------+
|64                                                                     |Forwarded
|
|128                                                                    |Dropped
|
|192                                                                    |Consumed
|
Limit Reached
Query terminated
```

4. create stream from databus_interfaces

```
CREATE stream stream_databus_interfaces
(iface_in_key VARCHAR ,iface_out_key VARCHAR, label VARCHAR, IfName VARCHAR,
IfIndex BIGINT)
WITH (kafka_topic='databus_interfaces', partitions=1, value_format='json');
```

4.1 rekey stream_databus_interfaces to stream_databus_interfaces_in by iface_in_key

```
CREATE stream stream_databus_interfaces_in AS
 SELECT * FROM stream_databus_interfaces \
 PARTITION BY iface_in_key;
```

4.2 rekey stream_databus_interfaces to stream_databus_interfaces_out by iface_out_key

```
CREATE stream stream_databus_interfaces_out AS
 SELECT * FROM stream_databus_interfaces \
 PARTITION BY iface_out_key;
```

4.3 create table table_databus_interfaces_in

```
CREATE TABLE table_databus_interfaces_in
(iface_in_key VARCHAR PRIMARY KEY ,iface_out_key VARCHAR, label VARCHAR, IfName
VARCHAR, IfIndex BIGINT)
WITH (
     KAFKA_TOPIC = 'STREAM_DATABUS_INTERFACES_IN',
     VALUE_FORMAT = 'JSON'
   );
```

4.4 create table table_databus_interfaces_in

```
CREATE TABLE table_databus_interfaces_out
(iface_in_key VARCHAR ,iface_out_key VARCHAR PRIMARY KEY, label VARCHAR, IfName
VARCHAR, IfIndex BIGINT)
WITH (
     KAFKA_TOPIC = 'STREAM_DATABUS_INTERFACES_OUT',
     VALUE_FORMAT = 'JSON'
   );
```

4.4 Verification

```
ksql> select * from TABLE_DATABUS_INTERFACES_IN where label = 'lab-erta-ndc' EMIT
CHANGES limit 1;
+--------------------------+--------------------------+--------------------------
+--------------------------+--------------------------+
|IFACE_IN_KEY              |IFACE_OUT_KEY             |LABEL
|IFNAME                    |IFINDEX                   |
+--------------------------+--------------------------+--------------------------
+--------------------------+--------------------------+
|1_lab-erta-ndc            |1_lab-erta-ndc            |lab-erta-ndc
|system                    |1                         |
Limit Reached
Query terminated
ksql> select * from TABLE_DATABUS_INTERFACES_OUT where label = 'lab-erta-ndc' EMIT
CHANGES limit 1;
+--------------------------+--------------------------+--------------------------
+--------------------------+--------------------------+
|IFACE_OUT_KEY             |IFACE_IN_KEY              |LABEL
|IFNAME                    |IFINDEX                   |
+--------------------------+--------------------------+--------------------------
+--------------------------+--------------------------+
|1_lab-erta-ndc            |1_lab-erta-ndc            |lab-erta-ndc
|system                    |1                         |
Limit Reached
Query terminated
```

At this stage we are ready to join our stream "STREAM_DATA_HOST_RAW_1M_REKEY" with 3 tables "TABLE_DATABUS_INTERFACES_IN" "TABLE_DATABUS_INTERFACES_OUT" and "TABLE_FORWARDING_STATUS"

5. Enrich stream

```
ksql>
SELECT
     STREAM_DATA_HOST_RAW_1M_REKEY.iface_in as iface_in,
     TABLE_DATABUS_INTERFACES_IN.ifName as iface_in_name,
     STREAM_DATA_HOST_RAW_1M_REKEY.iface_out as iface_out,
     TABLE_DATABUS_INTERFACES_OUT.ifName as iface_out_name,
     TABLE_FORWARDING_STATUS.LABEL as forwarding_status_name,
     STREAM_DATA_HOST_RAW_1M_REKEY.CUSTOM_PRIMITIVES['forwarding_status'] as
forwarding_status
   FROM STREAM_DATA_HOST_RAW_1M_REKEY
    INNER JOIN TABLE_DATABUS_INTERFACES_IN ON
     STREAM_DATA_HOST_RAW_1M_REKEY.iface_in_key =
TABLE_DATABUS_INTERFACES_IN.iface_in_key
    INNER JOIN TABLE_DATABUS_INTERFACES_OUT ON
     STREAM_DATA_HOST_RAW_1M_REKEY.iface_out_key =
TABLE_DATABUS_INTERFACES_OUT.iface_OUT_key
    INNER JOIN TABLE_FORWARDING_STATUS ON
     STREAM_DATA_HOST_RAW_1M_REKEY.CUSTOM_PRIMITIVES['forwarding_status'] =
TABLE_FORWARDING_STATUS.id
   EMIT CHANGES LIMIT 10;
```

5.1 Verification

```
+---------------------+---------------------+---------------------+---------------
------+---------------------+---------------------+
|IFACE_IN             |IFACE_IN_NAME        |IFACE_OUT            |IFACE_OUT_NAME
|FORWARDING_STATUS_NAM|FORWARDING_STATUS    |
|                     |                     |                     |
|E                    |                     |
+---------------------+---------------------+---------------------+---------------
------+---------------------+---------------------+
|10                   |To-VPRN403-hairpin   |5                    |to-mate
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |7                    |to-obr
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |7                    |to-obr
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |7                    |to-obr
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |7                    |to-obr
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |5                    |to-mate
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |7                    |to-obr
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |7                    |to-obr
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |7                    |to-obr
```

```
|Forwarded            |64                   |
|10                   |To-VPRN403-hairpin   |7                    |to-obr
|Forwarded            |64                   |
```