New IP Who Dis?

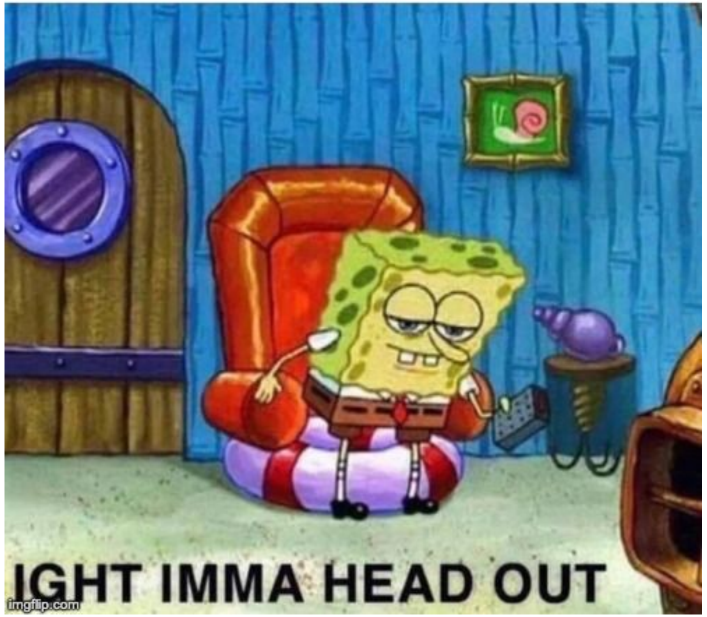
Exploring AWS VPC Flow Logs with Immerse and OmniSciDB

Tom Bowyer
Security Engineer- OmniSci

Overview

- What is an AWS VPC Flow Log
- How do I turn it on
- What Information can I get from these logs
- QA

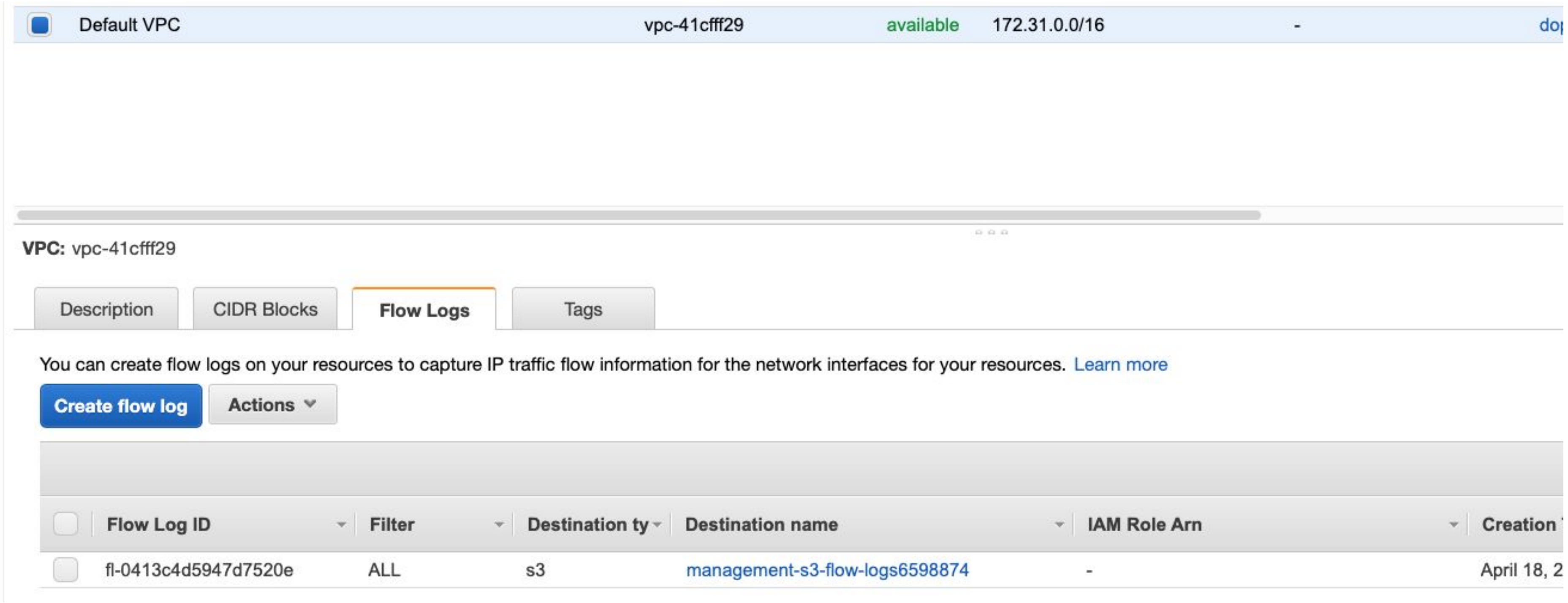
THIS TALK STARTS



What is an AWS VPC flow log

- Kinda like Netflow
- Contains all VPC network transit data
- Stores the data in S3 as a csv…..can also go to cloud watch if you hate yourself.

How to turn on



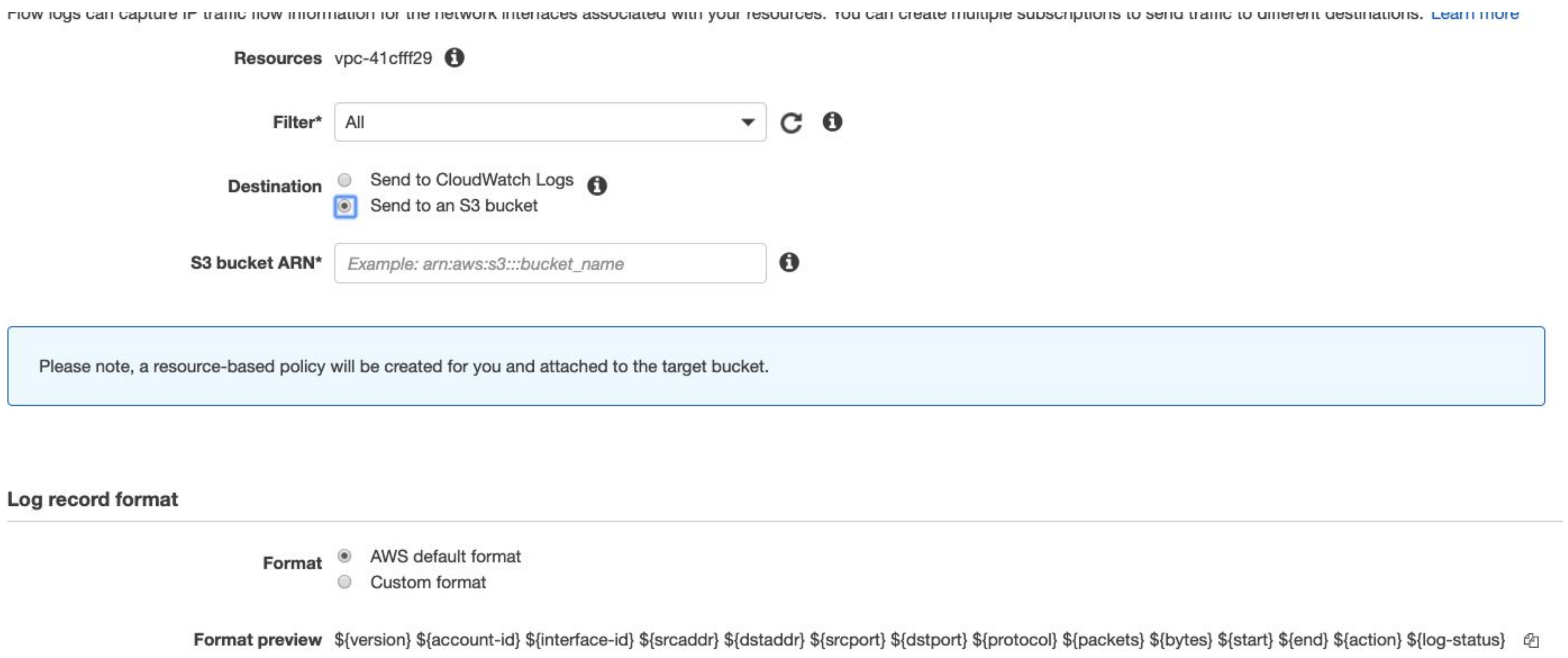
	Default VPC	vpc-41cfff29	available	172.31.0.0/16	-	do

VPC: vpc-41cfff29

You can create flow logs on your resources to capture IP traffic flow information for the network interfaces for your resources. Learn more

Flow Log ID	Filter	Destination ty	Destination name	IAM Role Arn	Creation
fl-0413c4d5947d7520e	ALL	s3	management-s3-flow-logs6598874	-	April 18, 2

How to turn on



Flow logs can capture IP traffic flow information for the network interfaces associated with your resources. You can create multiple subscriptions to send traffic to different destinations. Learn more

Resources	vpc-41cfff29
Filter*	All
Destination	Send to CloudWatch Logs [✓ Send to an S3 bucket]
S3 bucket ARN*	*Example: arn:aws:s3:::bucket_name*

Please note, a resource-based policy will be created for you and attached to the target bucket.

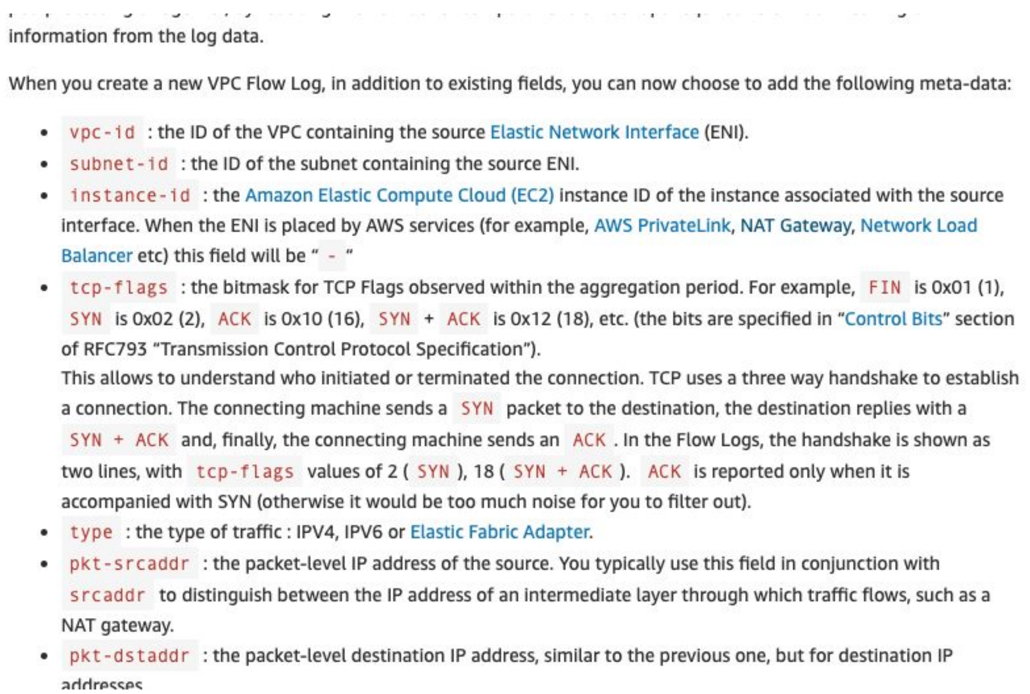
Log record format

Format	[✓ AWS default format] Custom format
Format preview	${version} ${account-id} ${interface-id} ${srcaddr} ${dstaddr} ${srcport} ${dstport} ${protocol} ${packets} ${bytes} ${start} ${end} ${action} ${log-status}

account-id	Your AWS Account ID
interface-id	The ID of the network interface for which the traffic is recorded.
srcaddr	The source address
dstaddr	The Destination Address
srcport	The Source port
dstport	The Destination Port
protocol	What Protocol number
packets	Number of packets during the flow
bytes	Bytes Transferred during the flow
start	Start time of the flow (Unix)
end	End time of the flow (Unix)
action	Accepted or Rejected

https://docs.aws.amazon.com/vpc/latest/userguide/flow-logs.html#flow-logs-fields

Additional metadata added just last month. Requires a format change in your flowlog config :(.



information from the log data.

When you create a new VPC Flow Log, in addition to existing fields, you can now choose to add the following meta-data:

- `vpc-id` : the ID of the VPC containing the source Elastic Network Interface (ENI).
- `subnet-id` : the ID of the subnet containing the source ENI.
- `instance-id` : the Amazon Elastic Compute Cloud (EC2) instance ID of the instance associated with the source interface. When the ENI is placed by AWS services (for example, AWS PrivateLink, NAT Gateway, Network Load Balancer etc) this field will be " `-` "
- `tcp-flags` : the bitmask for TCP Flags observed within the aggregation period. For example, `FIN` is 0x01 (1), `SYN` is 0x02 (2), `ACK` is 0x10 (16), `SYN` + `ACK` is 0x12 (18), etc. (the bits are specified in "Control Bits" section of RFC793 "Transmission Control Protocol Specification").
  This allows to understand who initiated or terminated the connection. TCP uses a three way handshake to establish a connection. The connecting machine sends a `SYN` packet to the destination, the destination replies with a `SYN + ACK` and, finally, the connecting machine sends an `ACK` . In the Flow Logs, the handshake is shown as two lines, with `tcp-flags` values of 2 (`SYN`), 18 (`SYN + ACK`). `ACK` is reported only when it is accompanied with SYN (otherwise it would be too much noise for you to filter out).
- `type` : the type of traffic : IPV4, IPV6 or Elastic Fabric Adapter.
- `pkt-srcaddr` : the packet-level IP address of the source. You typically use this field in conjunction with `srcaddr` to distinguish between the IP address of an intermediate layer through which traffic flows, such as a NAT gateway.
- `pkt-dstaddr` : the packet-level destination IP address, similar to the previous one, but for destination IP

Now What

THAT FACE YOU MAKE WHEN
YOU DON'T CARE BUT DON'T WAN'T TO BE RUDE
trollomat.com

ETL

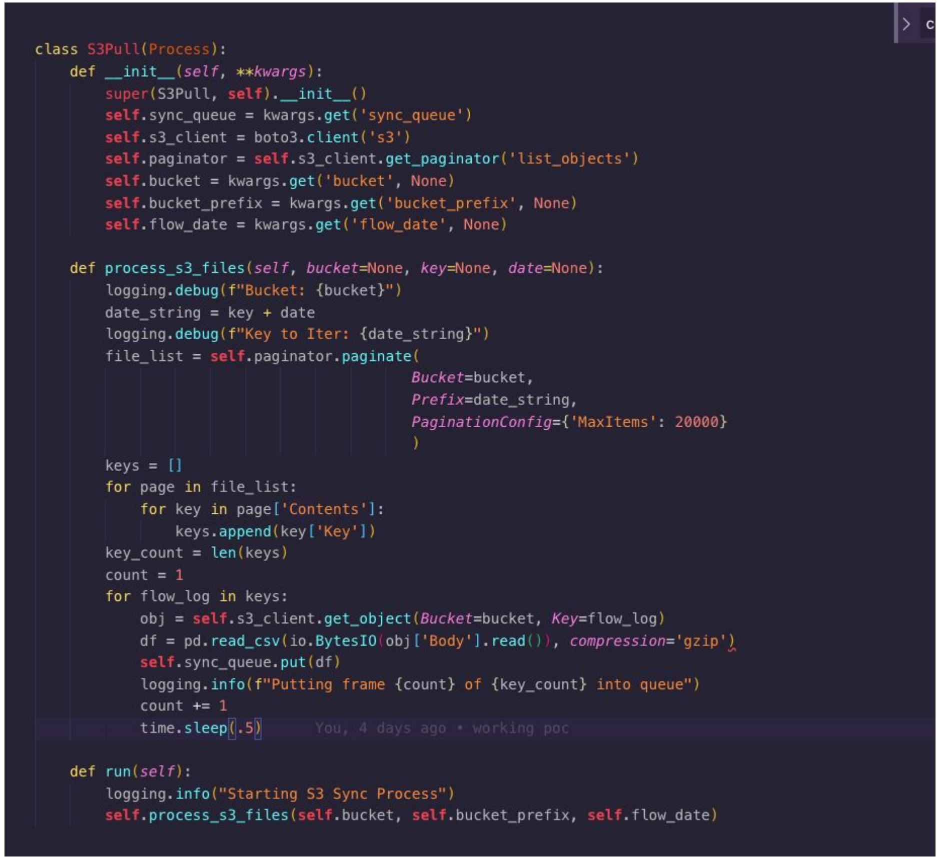
- Add Geoenrichment
  - Need me some countries
- Add threat intel
  - OTX
- Fast import and fully automated
- Python very much

https://github.com/Zeerg/Conference-Talks/tree/master/omnisci-converge-2019/s3flow_sync

Extract

Multiprocess step 1

- Pull data from S3 and insert into queue



```python
class S3Pull(Process):
    def __init__(self, **kwargs):
        super(S3Pull, self).__init__()
        self.sync_queue = kwargs.get('sync_queue')
        self.s3_client = boto3.client('s3')
        self.paginator = self.s3_client.get_paginator('list_objects')
        self.bucket = kwargs.get('bucket', None)
        self.bucket_prefix = kwargs.get('bucket_prefix', None)
        self.flow_date = kwargs.get('flow_date', None)

    def process_s3_files(self, bucket=None, key=None, date=None):
        logging.debug(f"Bucket: {bucket}")
        date_string = key + date
        logging.debug(f"Key to Iter: {date_string}")
        file_list = self.paginator.paginate(
                                           Bucket=bucket,
                                           Prefix=date_string,
                                           PaginationConfig={'MaxItems': 20000}
                                           )
        keys = []
        for page in file_list:
            for key in page['Contents']:
                keys.append(key['Key'])
        key_count = len(keys)
        count = 1
        for flow_log in keys:
            obj = self.s3_client.get_object(Bucket=bucket, Key=flow_log)
            df = pd.read_csv(io.BytesIO(obj['Body'].read()), compression='gzip')
            self.sync_queue.put(df)
            logging.info(f"Putting frame {count} of {key_count} into queue")
            count += 1
            time.sleep(.5)      You, 4 days ago • working poc

    def run(self):
        logging.info("Starting S3 Sync Process")
        self.process_s3_files(self.bucket, self.bucket_prefix, self.flow_date)
```

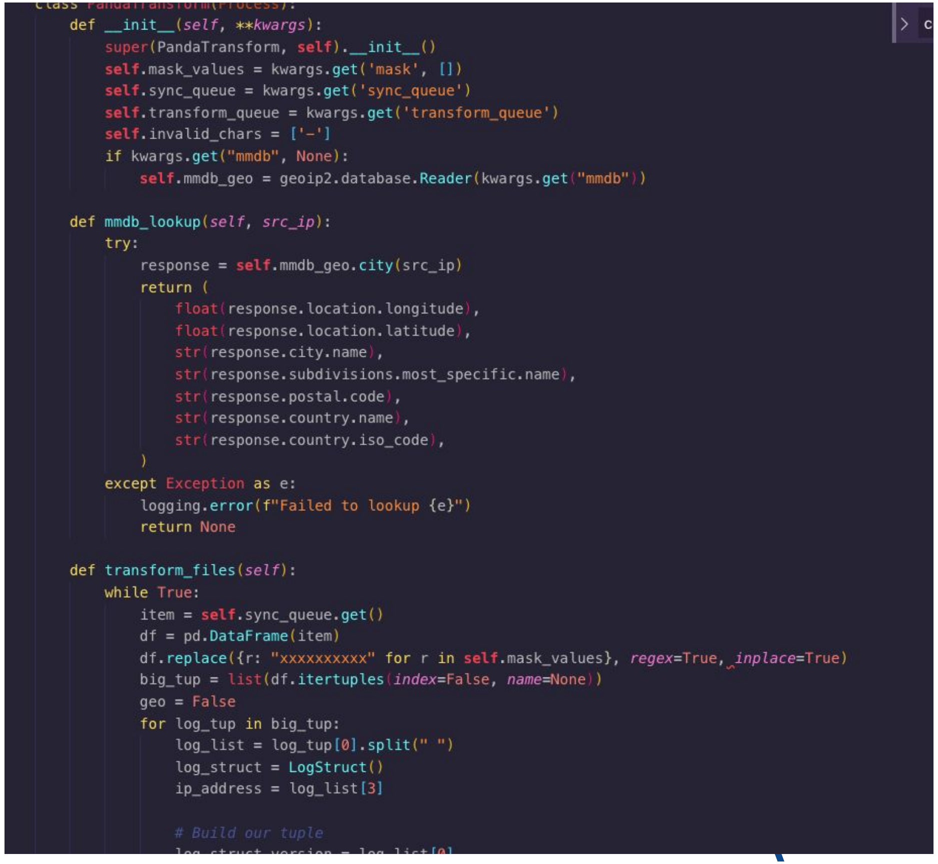
Transform(ers)

Multiprocess step 2

- Pull from queue then add dem countries using maxmind
- Also mask things we don't want to share

- Add threat intel lookup

- Add to queue

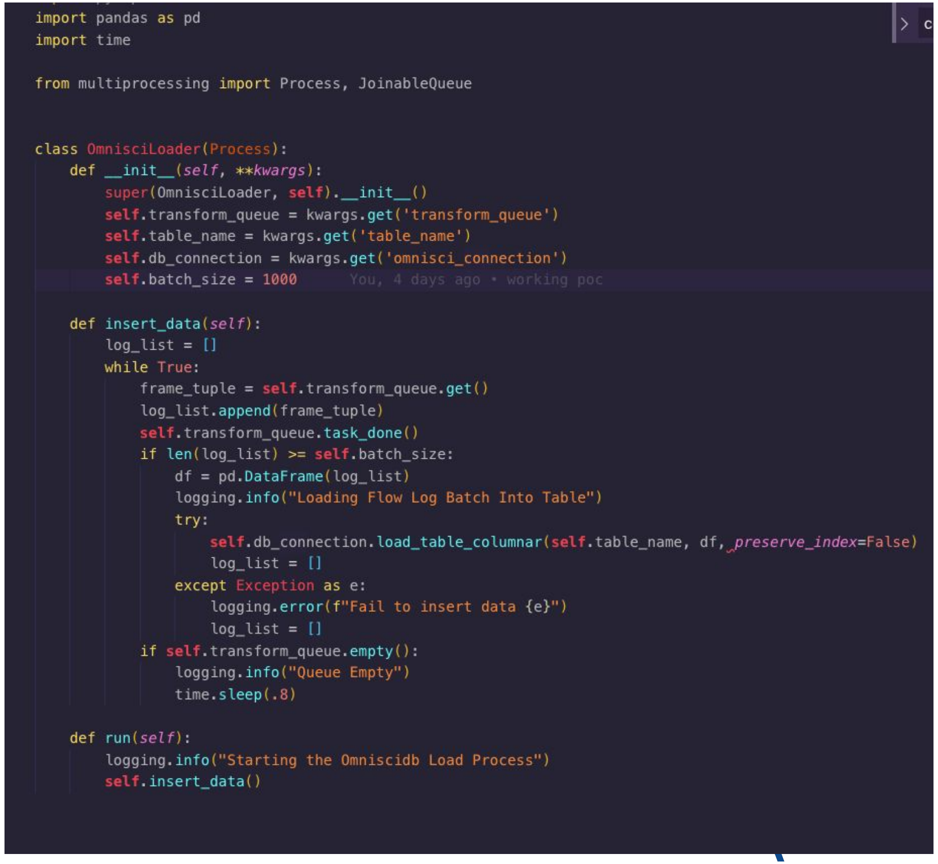


```
class PandaTransform(Process):
    def __init__(self, **kwargs):
        super(PandaTransform, self).__init__()
        self.mask_values = kwargs.get('mask', [])
        self.sync_queue = kwargs.get('sync_queue')
        self.transform_queue = kwargs.get('transform_queue')
        self.invalid_chars = ['-']
        if kwargs.get("mmdb", None):
            self.mmdb_geo = geoip2.database.Reader(kwargs.get("mmdb"))

    def mmdb_lookup(self, src_ip):
        try:
            response = self.mmdb_geo.city(src_ip)
            return (
                float(response.location.longitude),
                float(response.location.latitude),
                str(response.city.name),
                str(response.subdivisions.most_specific.name),
                str(response.postal.code),
                str(response.country.name),
                str(response.country.iso_code),
            )
        except Exception as e:
            logging.error(f"Failed to lookup {e}")
            return None

    def transform_files(self):
        while True:
            item = self.sync_queue.get()
            df = pd.DataFrame(item)
            df.replace({r: "xxxxxxxxxx" for r in self.mask_values}, regex=True, inplace=True)
            big_tup = list(df.itertuples(index=False, name=None))
            geo = False
            for log_tup in big_tup:
                log_list = log_tup[0].split(" ")
                log_struct = LogStruct()
                ip_address = log_list[3]

                # Build our tuple
                log_struct.version = log_list[0]
```

Load

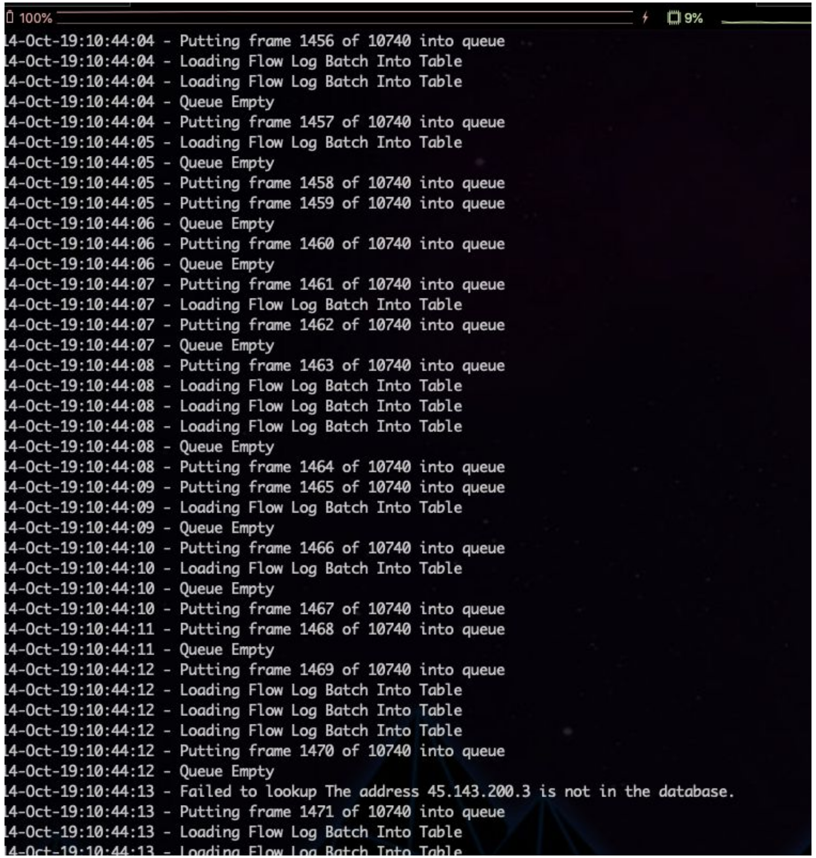
Multiprocess step 3

- Grab from load queue and...
- Load them frames



```
import pandas as pd
import time

from multiprocessing import Process, JoinableQueue


class OmnisciLoader(Process):
    def __init__(self, **kwargs):
        super(OmnisciLoader, self).__init__()
        self.transform_queue = kwargs.get('transform_queue')
        self.table_name = kwargs.get('table_name')
        self.db_connection = kwargs.get('omnisci_connection')
        self.batch_size = 1000      You, 4 days ago • working poc

    def insert_data(self):
        log_list = []
        while True:
            frame_tuple = self.transform_queue.get()
            log_list.append(frame_tuple)
            self.transform_queue.task_done()
            if len(log_list) >= self.batch_size:
                df = pd.DataFrame(log_list)
                logging.info("Loading Flow Log Batch Into Table")
                try:
                    self.db_connection.load_table_columnar(self.table_name, df, preserve_index=False)
                    log_list = []
                except Exception as e:
                    logging.error(f"Fail to insert data {e}")
                    log_list = []
            if self.transform_queue.empty():
                logging.info("Queue Empty")
                time.sleep(.8)

    def run(self):
        logging.info("Starting the Omniscidb Load Process")
        self.insert_data()
```

ETL



```
100%                                                              9%
14-Oct-19:10:44:04 - Putting frame 1456 of 10740 into queue
14-Oct-19:10:44:04 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:04 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:04 - Queue Empty
14-Oct-19:10:44:04 - Putting frame 1457 of 10740 into queue
14-Oct-19:10:44:05 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:05 - Queue Empty
14-Oct-19:10:44:05 - Putting frame 1458 of 10740 into queue
14-Oct-19:10:44:05 - Putting frame 1459 of 10740 into queue
14-Oct-19:10:44:06 - Queue Empty
14-Oct-19:10:44:06 - Putting frame 1460 of 10740 into queue
14-Oct-19:10:44:06 - Queue Empty
14-Oct-19:10:44:07 - Putting frame 1461 of 10740 into queue
14-Oct-19:10:44:07 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:07 - Putting frame 1462 of 10740 into queue
14-Oct-19:10:44:07 - Queue Empty
14-Oct-19:10:44:08 - Putting frame 1463 of 10740 into queue
14-Oct-19:10:44:08 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:08 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:08 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:08 - Queue Empty
14-Oct-19:10:44:08 - Putting frame 1464 of 10740 into queue
14-Oct-19:10:44:09 - Putting frame 1465 of 10740 into queue
14-Oct-19:10:44:09 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:09 - Queue Empty
14-Oct-19:10:44:10 - Putting frame 1466 of 10740 into queue
14-Oct-19:10:44:10 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:10 - Queue Empty
14-Oct-19:10:44:10 - Putting frame 1467 of 10740 into queue
14-Oct-19:10:44:11 - Putting frame 1468 of 10740 into queue
14-Oct-19:10:44:11 - Queue Empty
14-Oct-19:10:44:12 - Putting frame 1469 of 10740 into queue
14-Oct-19:10:44:12 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:12 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:12 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:12 - Putting frame 1470 of 10740 into queue
14-Oct-19:10:44:12 - Queue Empty
14-Oct-19:10:44:13 - Failed to lookup The address 45.143.200.3 is not in the database.
14-Oct-19:10:44:13 - Putting frame 1471 of 10740 into queue
14-Oct-19:10:44:13 - Loading Flow Log Batch Into Table
14-Oct-19:10:44:13 - Loading Flow Log Batch Into Table
```

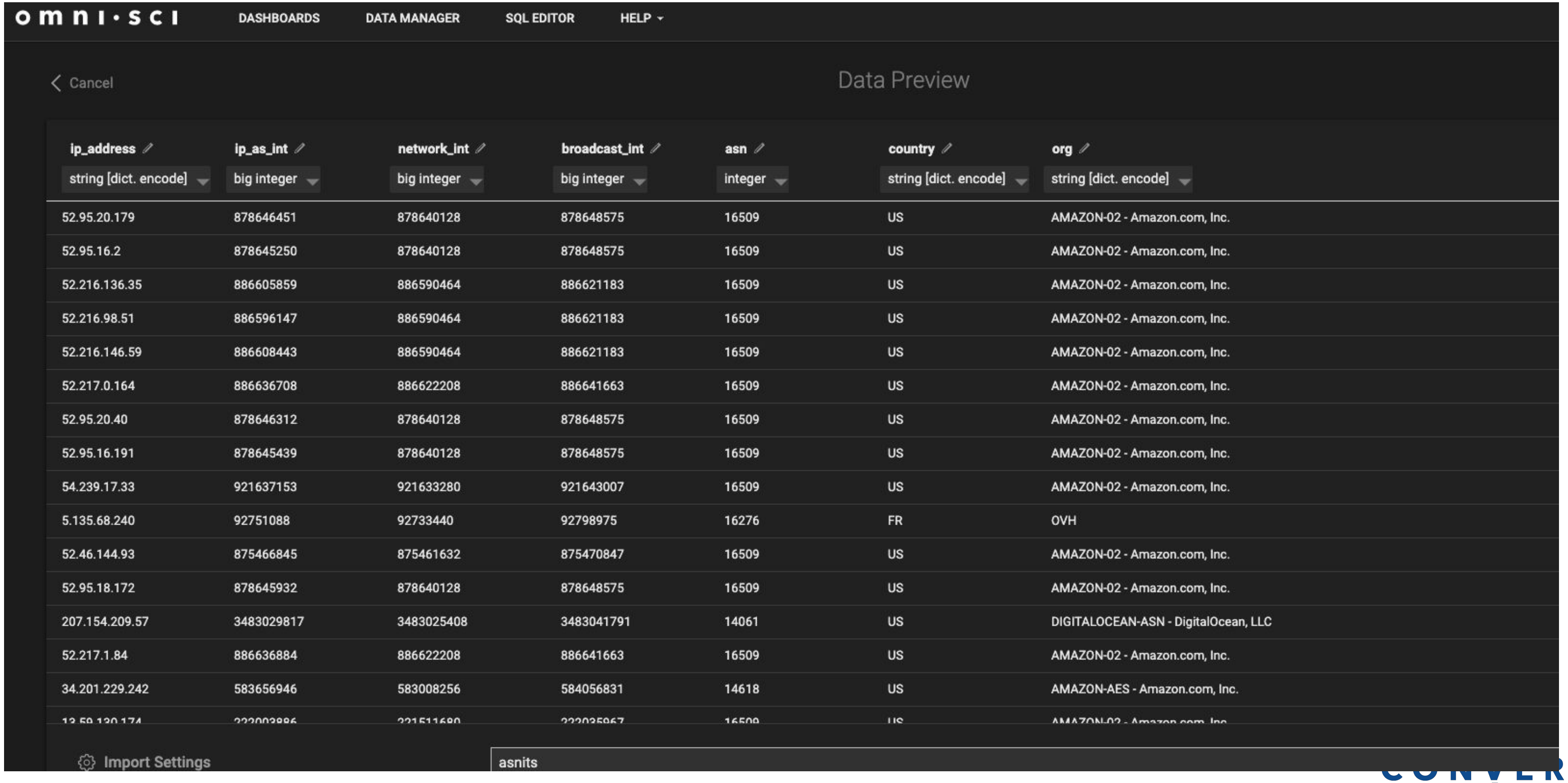
https://github.com/Zeerg/Conference-Talks/tree/master/omnisci-converge-2019/s3flow_sync

What about the ASN?

- ?????? uhhhhh
- ETL 150 million records again adding 1 column
- Or
- Do a join on a table that has ASNs and IPs

WHEN YOU DON'T HAVE TIME TO ETL 150MILLION ROWS
Improvise. Adapt. Overcome
imgflip.com

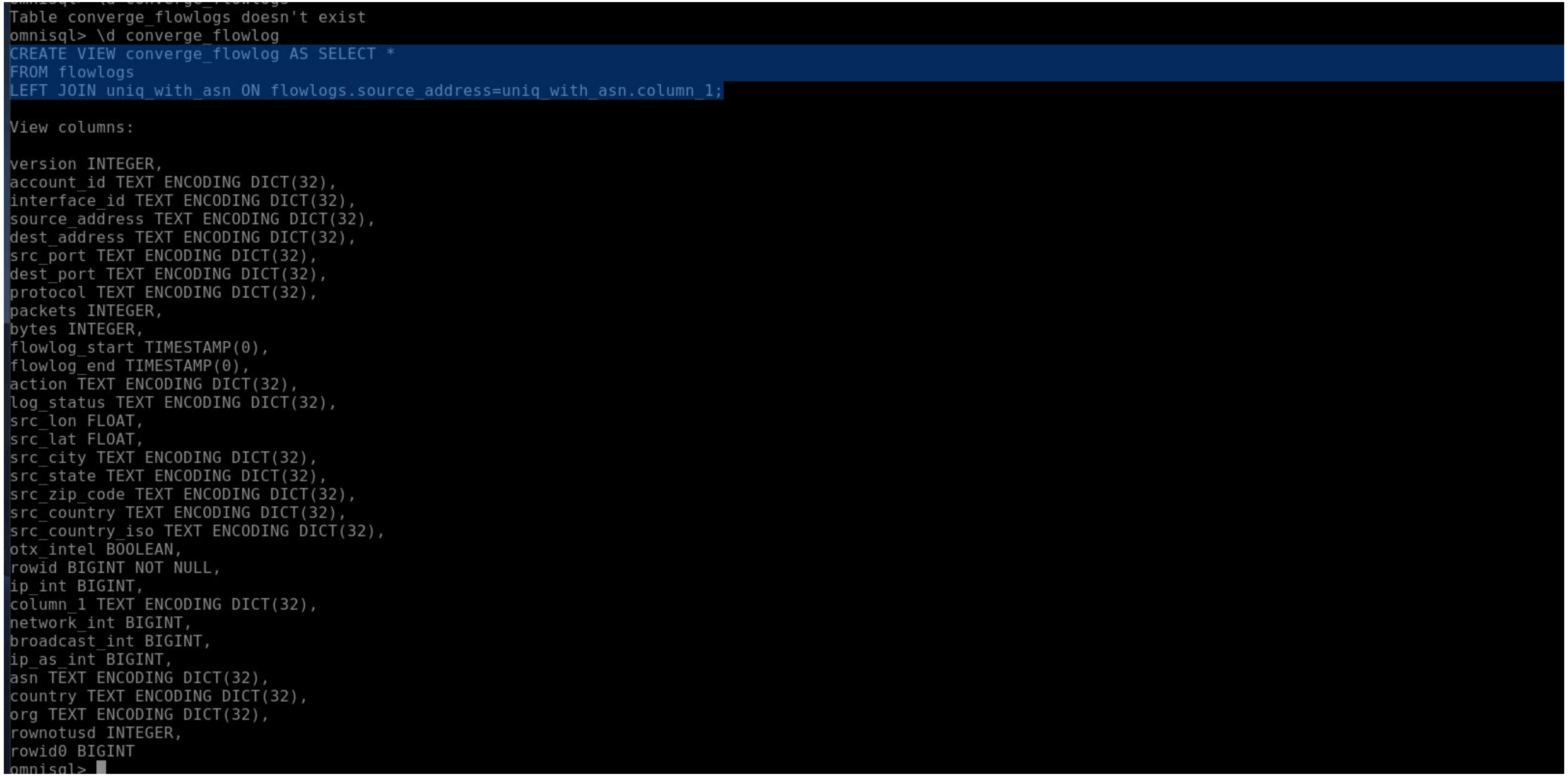
What about the ASN?



ip_address	ip_as_int	network_int	broadcast_int	asn	country	org
string [dict. encode]	big integer	big integer	big integer	integer	string [dict. encode]	string [dict. encode]
52.95.20.179	878646451	878640128	878648575	16509	US	AMAZON-02 - Amazon.com, Inc.
52.95.16.2	878645250	878640128	878648575	16509	US	AMAZON-02 - Amazon.com, Inc.
52.216.136.35	886605859	886590464	886621183	16509	US	AMAZON-02 - Amazon.com, Inc.
52.216.98.51	886596147	886590464	886621183	16509	US	AMAZON-02 - Amazon.com, Inc.
52.216.146.59	886608443	886590464	886621183	16509	US	AMAZON-02 - Amazon.com, Inc.
52.217.0.164	886636708	886622208	886641663	16509	US	AMAZON-02 - Amazon.com, Inc.
52.95.20.40	878646312	878640128	878648575	16509	US	AMAZON-02 - Amazon.com, Inc.
52.95.16.191	878645439	878640128	878648575	16509	US	AMAZON-02 - Amazon.com, Inc.
54.239.17.33	921637153	921633280	921643007	16509	US	AMAZON-02 - Amazon.com, Inc.
5.135.68.240	92751088	92733440	92798975	16276	FR	OVH
52.46.144.93	875466845	875461632	875470847	16509	US	AMAZON-02 - Amazon.com, Inc.
52.95.18.172	878645932	878640128	878648575	16509	US	AMAZON-02 - Amazon.com, Inc.
207.154.209.57	3483029817	3483025408	3483041791	14061	US	DIGITALOCEAN-ASN - DigitalOcean, LLC
52.217.1.84	886636884	886622208	886641663	16509	US	AMAZON-02 - Amazon.com, Inc.
34.201.229.242	583656946	583008256	584056831	14618	US	AMAZON-AES - Amazon.com, Inc.
13.59.130.174	222003886	221511680	222035967	16509	US	AMAZON-02 - Amazon.com, Inc.

asnits

What about the ASN?

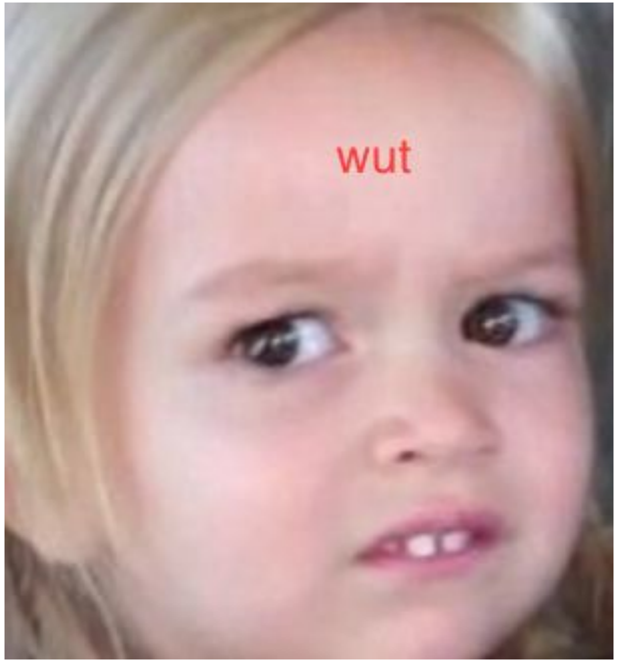


```
Table converge_flowlogs doesn't exist
omnisql> \d converge_flowlog
CREATE VIEW converge_flowlog AS SELECT *
FROM flowlogs
LEFT JOIN uniq_with_asn ON flowlogs.source_address=uniq_with_asn.column_1;

View columns:

version INTEGER,
account_id TEXT ENCODING DICT(32),
interface_id TEXT ENCODING DICT(32),
source_address TEXT ENCODING DICT(32),
dest_address TEXT ENCODING DICT(32),
src_port TEXT ENCODING DICT(32),
dest_port TEXT ENCODING DICT(32),
protocol TEXT ENCODING DICT(32),
packets INTEGER,
bytes INTEGER,
flowlog_start TIMESTAMP(0),
flowlog_end TIMESTAMP(0),
action TEXT ENCODING DICT(32),
log_status TEXT ENCODING DICT(32),
src_lon FLOAT,
src_lat FLOAT,
src_city TEXT ENCODING DICT(32),
src_state TEXT ENCODING DICT(32),
src_zip_code TEXT ENCODING DICT(32),
src_country TEXT ENCODING DICT(32),
src_country_iso TEXT ENCODING DICT(32),
otx_intel BOOLEAN,
rowid BIGINT NOT NULL,
ip_int BIGINT,
column_1 TEXT ENCODING DICT(32),
network_int BIGINT,
broadcast_int BIGINT,
ip_as_int BIGINT,
asn TEXT ENCODING DICT(32),
country TEXT ENCODING DICT(32),
org TEXT ENCODING DICT(32),
rownotusd INTEGER,
rowid0 BIGINT
omnisql>
```

Now We Explore

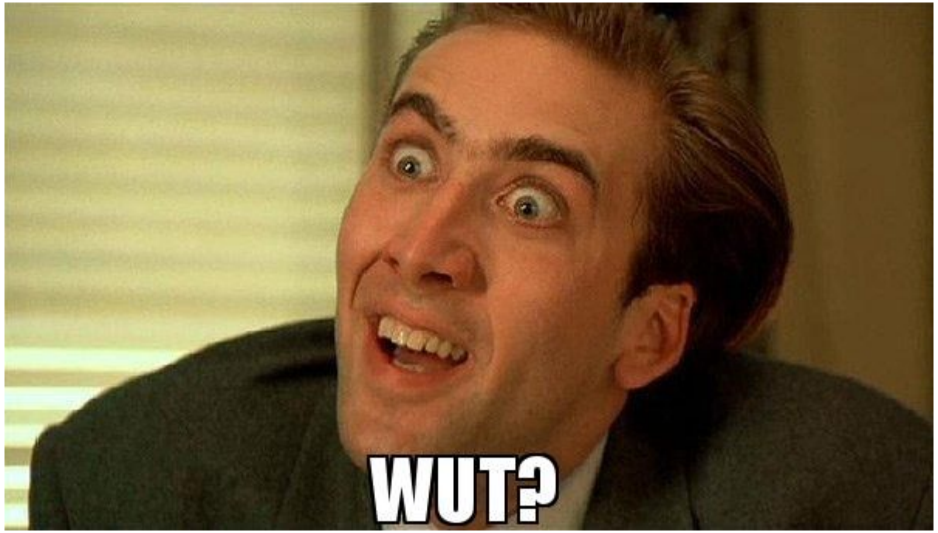
Improving Security Posture

- Where to begin?
- Hey we need to limit outbound ports for this awesome compliance requirement.
- or really removing 0.0.0.0/0 outbound



Threat Hunting

- Checking for allowed SSH
- Are we using port 80 inbound still?
- Is MySQL open to the world
- Threat Hunting
- Do I have hosts that allow ping



Pro Tip

- If network ACLs attached to a NAT gateway don't explicitly deny traffic from the internet, internet traffic to the NAT gateway appears accepted.
- However, the actual traffic isn't accepted by the NAT gateway and is dropped.
- Heart Attack No More!

CONVERGE by omni·sci

Links

- ASN List
https://iptoasn.com/

- Maxmind
https://www.maxmind.com/en/geoip2-city
- ETL Tool
https://github.com/Zeerg/Conference-Talks/tree/master/omnisci-converge-2019/s3flow_sync

Q&A

CONVERGE by omni·sci

thank you