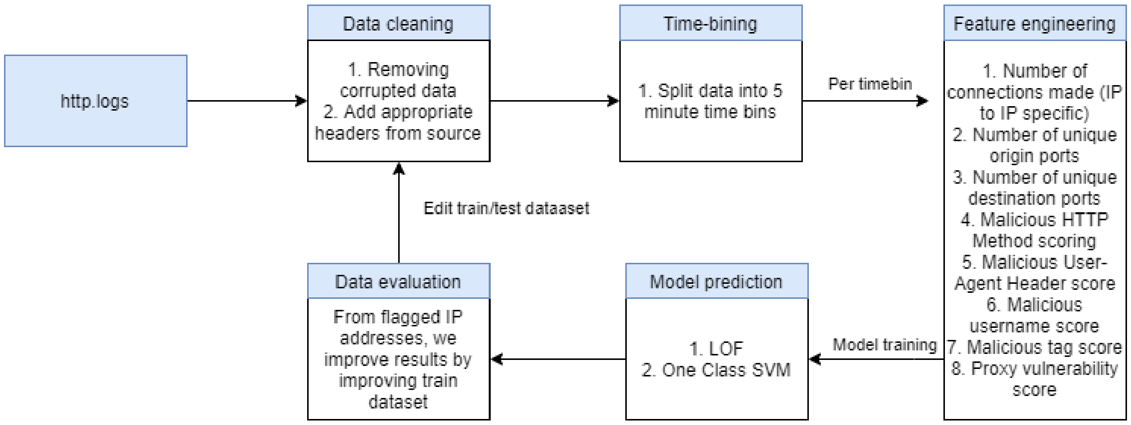
1 Network Reconnaissance Detection Algorithm

Author: Khor Zile @ Khor Zi Long
June 2020

1.1 Methodology

The overall algorithm architecture employed in reconnaissance detection exercise.


http.logs
Data cleaning
1. Removing corrupted data
2. Add appropriate headers from source
Time-bining
1. Split data into 5 minute time bins
Per timebin
Feature engineering
1. Number of connections made (IP to IP specific)
2. Number of unique origin ports
3. Number of unique destination ports
4. Malicious HTTP Method scoring
5. Malicious User-Agent Header score
6. Malicious username score
7. Malicious tag score
8. Proxy vulnerability score
Model training
Model prediction
1. LOF
2. One Class SVM
Data evaluation
From flagged IP addresses, we improve results by improving train dataset
Edit train/test dataaset

Network reconnaissance is a huge field involving various methods, such as port scanning, sub-domain exploration, OS fingerprinting and et cetera. In this report, the focus of the algorithm is to parse HTTP logs and potentially identify IP addresses that are conducting activities associated with network reconnaissance.

Since this dataset was obtained from a pentesting event held by Mid-Atlantic CCDC in 2012, the dataset contains a combination of network reconnaissance and actual attacks against the network. Due to the unlabelled(*) nature of the dataset, it is impossible to identify which row is logged from an reconnaissance attempt or from an actual attack.

Combining ideas from [1] and [2], we attempt to combine both rule-based and ML approaches to reconnaissance detection. We include a *rule-based* approach by generating features that would double as rule-based approach to generating alerts. The features incorporates rules based on the scoring system, where known properties of widely used reconnaissance tools are given a higher (more malicious) score. When fed into the ML model, this model learns both these features and scores, eventually learning from these rules.

Although the features engineered, as explained in the *feature engineering* section, is built with network reconnaissance in mind, it is inevitable to detect network attacks as both are anomalous HTTP traffic. As such, I recognize that this algorithm would pick up not just reconnaissance but actual attacks as well.

The following report is broken down into the separate architecture components, starting with data pre-processing, time-bining, feature engineering, running the model, data evaluation and retraining the model.

(*) Disclaimer: The source providing this set of HTTP logs does provide a snort analysis of the network logs. This includes analysis includes alerts flagged out based on the Snort rule-based alerts [1]. However, I experienced technical difficulties in running Snort software on my computer. It is possible to use these alerts as a base for tagging the dataset and run semi-supervised learning to obtain better results in the future.

1.2 Data pre-processing

This section is largely focused on cleaning and formatting the data as follows: 1. Importing the necessary libraries 2. Setting up the pandas dataframe and include missing headers(*) 3. Removing poorly formatted rows

One of the observations from processing the dataset provided was that the delimiter used (default for Bro logs is ‘$\hat{}$’) was inappropriate for this dataset as certain payload included the ‘$\hat{}$’. we remove these rows to ensure that our data is properly formatted before moving onto feature engineering and model training. As these rows likely included malicious payloads that caused the formatting for a particular and subsequent rows to be corrupted, dropping them might cause some critical information to be lost. This can be further considered in subsequent iterations of this project

From the warnings thrown by parsing the dataset, we observe that columns “referrer”, “user_agent” and “request_body_len” contain corrupted cell values. We then search for the delimiter ‘$\hat{}$’ that would still be present in the data (since it was not caught as a delimiter) and other possible indicators of corruption.

Further improvements

1. Consider labelling dropped rows as malicious

(*) headers are obtained from 4

```
[1]: import pandas as pd
     import numpy as np
     import os
     import sklearn
     headers = [
         'ts', 'uid', 'id.orig_h', 'id.orig_p', 'id.resp_h', 'id.resp_p',
         'trans_depth', 'method', 'host', 'uri', 'referrer', 'user_agent',
         'request_body_len', 'response_body_len', 'status_code',
         'status_msg', 'info_code', 'info_msg', 'filename', 'tags',
         'username', 'password', 'proxied', 'orig_fuids',
         'orig_mine_types', 'resp_fuids', 'resp_mime_types'
     ]
     headers_with_features = headers + ["method_score", "ua_score", "username_score",␣
      ↪"tag_score", "proxied_score"]
     pd.set_option('max_columns', None)
```

```
[2]: http_logs = pd.read_csv('http.log', sep='\t', names=headers, error_bad_lines =␣
      ↪False)
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\IPython\core\interactiveshell.py:3063: DtypeWarning: Columns (12,13,14)
```

```
have mixed types.Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

[3]:
```
## clean data
## from the warnings after parsing dataset, we know "referrer",
## "user_agent" and "request_body_len" columns contain bad rows

## remove rows from "referrer" that contains the delimiter '\t'
http_logs = http_logs[~http_logs.referrer.str.contains('\t')]
## remove rows from "user agent" that contains the delimiter '\t'
http_logs = http_logs[~http_logs.user_agent.str.contains('\t')]
## remove rows from "request_body_len" that contains the " " -- based on␣
 ↪observation
http_logs.request_body_len = http_logs.request_body_len.astype(str)
http_logs = http_logs[~http_logs.request_body_len.str.contains(" ", na=False)]
http_logs.request_body_len = pd.to_numeric(http_logs.request_body_len)

## convert ts to datetime format
http_logs.ts = pd.to_datetime(http_logs.ts, unit='s')
```

[4]:
```
print("Total rows left: {} compared to original dataset: 2048442".
 ↪format(http_logs.shape[0]))
print("Numbers of rows dropped: {}".format(2048442-http_logs.shape[0]))
```

```
Total rows left: 2047445 compared to original dataset: 2048442
Numbers of rows dropped: 997
```

[5]:
```
http_logs.to_csv("pre_processed_http_logs.csv", sep='\t', index=False)
```

1.3 Analysis of data and feature engineering

Dataset general information:

Time range: 16 March 2012, 1230H - 2047H The logs was obtained from a pentesting event held by Mid-Atlantic CCDC in 2012. The teams were students teams working for the Hospital of the East Collective (HEC); a collective of 8 regional hospitals.

In this section, we will analyse the available data from the logs and engineer features that will provide additional information for us to train and test our models on. The inspiration of these features are consolidated from various readings and general anomaly detection [1][2][3]. Extrapolating from the idea of anomaly detection, we narrowed down to features that best represent network reconnaissance. Mainly, domain exploration, port scanning and fingerprinting.

Below we describe the various features that we will be engineering.

1.4 Malicious scoring

Since the dataset is unlabelled, we will begin by converting log data into numerical values to facilitate model paramter training. To do so, we split headers that are non-numeric and assign a score to the different unique values in that column. All observations of unique values are recorded at the end of this notebook

1.4.1 HTTP Methods:

The below scoring is method is based on the following reading that lists down the differnte common HTTP and MDSN methods. This table is not inclusive of all unique values found. The score ranges from 0 (least severe) to 2 (most severe). The scores will then be normalized.

Do note that this table can be changed to include other key words when calculating score in the future. All values that do not fall in any of these categories are deemed malicious and automatically assigned to 2.

Known reconnaissance or malicious values like `meterpreter` (a metasploit reverse shell method) and `TESTZZZ` are immediately scored the highest, 2.

Score	Methods
0 (whitelist)	OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT,PROPFIND, PROPPATCH, MKCOL, COPY, MOVE, LOCK, UNLOCK,VERSIO-CONTROL, REPORT, CHECKOUT, CHECKIN, UNCHECKOUT, MKWORKSPACE, UPDATE, LABEL, MERGE, BASELINE-CONTROL, MKACTIVITY,ORDERPATCH,ACL,PATCH,SEARCH,BCOPY, BDELETE, BMOVE, BPROPFIND, BPROPPATCH, NOTIFY, POLL, SUBSCRIBE, UNSUBSCRIBE, X-MS-ENUMATTS
1	GNUTELLA, RPC_IN_DATA
2	meterpreter, TESTZZZ, everything else

1.4.2 User-Agent

There are many possible variations of User-Agents. Thus, it would be inefficient to score all permutations. However, we can blacklist keywords within the UA header and requests containing such keywords would be flagged as follows:

Score	Keywords
0	Everything else
1 (blacklist)	Nikto, Nmap, injection, Chucky12345678/1.0, Nikto, passwd, ../, sleep, waitfor, delay

Note that the above values are included from 'eyeballing' the data and picking out the most obvious efforts in reconnaissance and/or malicious activities. The first two Nikto and Nmap were included as both are widely used reconnaissance libraries.

1.4.3 Usernames

There were multiple attempts in injecting code via the `username` header. We shall make an attempt to whitelist common usernames that would seem to be non-malicious:

Score	Keywords
0 (whitelist)	-, customer, manager, user, guest
1	Everything else

1.4.4 Tags

In the tags header, two unique valeus were observed: `HTTP::URI_SQLI` and `(empty)`. SQL Injection is definitely an IOC and as such, we also score this column as follows:

Score	Keywords
0	(empty)
1	HTTP::URI_SQLI, everything else

1.4.5 Proxied

Eyeballing the unique values within the `proxied` column, there were numerous attempts at injecting code into the HTTP request. As there were too many requests, it would be overly-consuming to filter and find authentic proxy requests. A simple count of rows with proxied value - i.e. no proxied header accounted for 2046291 out of 2048442 rows. As such, we marked scored the column as follows:

Score	Keywords
0	-
1	Everything else

[6]:
```
## load csv to prevent running pre-processing numerous times
http_logs = pd.read_csv("pre_processed_http_logs.csv", sep='\t', error_bad_lines␣
 ↪= False)
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\IPython\core\interactiveshell.py:3063: DtypeWarning: Columns (14) have
mixed types.Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

[7]:
```
## create filter masks to create new columns

## sample logs to test on
sample_logs = http_logs.head(10000)

def method_score(row):
    whitelist = ["OPTIONS", "GET", "HEAD", "POST", "PUT", "DELETE", "TRACE",␣
↪"CONNECT", "PROPFIND",
                 "PROPPATCH", "MKCOL", "COPY", "MOVE", "LOCK", "UNLOCK",␣
↪"VERSIO-CONTROL", "REPORT",
                 "CHECKOUT", "CHECKIN", "UNCHECKOUT", "MKWORKSPACE", "UPDATE",␣
↪"LABEL", "MERGE",
                 "BASELINE-CONTROL", "MKACTIVITY", "ORDERPATCH", "ACL", "PATCH",␣
↪"SEARCH", "BCOPY",
                 "BDELETE", "BMOVE", "BPROPFIND", "BPROPPATCH", "NOTIFY",␣
↪"POLL", "SUBSCRIBE",
                 "UNSUBSCRIBE", "X-MS-ENUMATTS"]
    uncommon = ["GNUTELLA", "RPC_IN_DATA"]
    blacklist = ["meterpreter", "TESTZZZ"]
    default = 2
    if any(keyword in row.method for keyword in whitelist):
        return 0
    elif any(keyword in row.method for keyword in uncommon):
        return 1
    elif any(keyword in row.method for keyword in blacklist):
        return 2
    else:
        return default

def ua_score(row):
    blacklist = ["Nikto", "Nmap", "injection", "Chucky12345678/1.0", "passwd", ".
↪./", "sleep", "waitfor", "delay"]
    default = 0
    if any(keyword in row.user_agent for keyword in blacklist):
        return 1
    else:
        return default

def username_score(row):
    whitelist = ["-", "customer", "manager", "user", "guest"]
    default = 1
    if any(keyword in row.username for keyword in whitelist):
        return 0
    else:
        return default

def tag_score(row):
    blacklist = ["HTTP::URI_SQLI"]
    default = 0
    if any(keyword in row.tags for keyword in blacklist):
        return 1
    else:
        return default

def proxied_score(row):
    whitelist = ["-"]
    default = 1
    if any(keyword in row.proxied for keyword in whitelist):
        return 0
    else:
        return default

## Test for errors in mask
# print(sample_logs.apply(lambda row: method_score(row), axis=1).value_counts())
# print(sample_logs.apply(lambda row: ua_score(row), axis=1).value_counts())
# print(sample_logs.apply(lambda row: username_score(row), axis=1).
 ↪value_counts())
# print(sample_logs.apply(lambda row: tag_score(row), axis=1).value_counts())
# print(sample_logs.apply(lambda row: proxied_score(row), axis=1).value_counts())
```

[8]:
```
## apply all filter masks

print("--applying masks--")
http_logs['method_score'] = http_logs.apply(lambda row: method_score(row),␣
 ↪axis=1)
print("--method score done--")
http_logs["ua_score"] = http_logs.apply(lambda row: ua_score(row), axis=1)
print("--ua score done--")
http_logs["username_score"] = http_logs.apply(lambda row: username_score(row),␣
 ↪axis=1)
print("--username score done--")
http_logs["tag_score"] = http_logs.apply(lambda row: tag_score(row), axis=1)
print("--tag score done--")
http_logs["proxied_score"] = http_logs.apply(lambda row: proxied_score(row),␣
 ↪axis=1)
print("--proxied score done--")
```

```
--applying masks--
--method score done--
--ua score done--
--username score done--
--tag score done--
--proxied score done--
```

[9]:
```
http_logs.to_csv("features_added_http_logs.csv", sep='\t', index=False)
```

1.4.6 Timebin-ing data

Now, we that we have created our feature columns, we will now aggregate these scores based on 5 minute time bins. The reason a 5-minute time bin was selected is because it is the *longest* delay when using Nmap timing templates. This means that every time bin should see at least one reconnaissance packet. The 5-minute bining is done in the next section. In this section, we split the entire dataset into 1-hour-bins to and save these as separate files for file management purposes.

Also, since our dataset is unlabelled (*) we are just splitting the entire dataset into train (60%) and test (40%). There will not be any evaluation but we will observe which IP addresses are likely to be running reconnaissance.

This section involves: 1. Splitting the logs into train (60%) and test (40%). Since the data is unlabelled, we want a larger test dataset. 2. Splitting each train and test dataset by hour and saving them into separate files as such:

```
file directory
.
+-- basedir
|   +-- train
|   |   +-- train_*.csv
|   |   ...
|   +-- test
|   |   +-- test_*.csv
|   |   ...
where * is the hour of the day.
```

(*) As explained above, there is a possibility to run evaluation since the source [1] appears to have provided the alerts thrown by Snort rules. Theoretically, we can use this data as labels for each request packet logged and use it to evaluate our test results. However, I have been trying to run Snort on my local machine but to no avail and due to the limited time given for this project, I have put this as a stretch goal that can be completed.

[10]:
```
## load directly from saved features
http_logs = pd.read_csv('features_added_http_logs.csv', sep='\t',␣
 ↪error_bad_lines = False)
http_logs.ts = pd.to_datetime(http_logs.ts)
```

[11]:
```
from sklearn.model_selection import train_test_split
train, test = train_test_split(http_logs, train_size=0.6, shuffle=False)
print("train size: ", train.shape[0])
print("test size: ", test.shape[0])

if not os.path.exists("train"):
    os.mkdir("train")
if not os.path.exists("test"):
    os.mkdir("test")


for hour in range(24):
    res_train = train[train.ts.dt.hour == hour]
    res_test = test[test.ts.dt.hour == hour]

    if res_train.shape[0] == 0:
        pass
    else:
        res_train.to_csv("train/train_"+str(hour)+".csv", sep='\t', index=False)
    if res_test.shape[0] == 0:
        pass
    else:
        res_test.to_csv("test/test_"+str(hour)+".csv", sep='\t', index=False)

    print("-- finished splitting by {}H".format(hour))
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\model_selection\_split.py:2179: FutureWarning: From version
0.21, test_size will always complement train_size unless both are specified.
  FutureWarning)
```

```
train size:  1228467
test size:  818978
-- finished splitting by 0H
-- finished splitting by 1H
-- finished splitting by 2H
-- finished splitting by 3H
-- finished splitting by 4H
-- finished splitting by 5H
-- finished splitting by 6H
-- finished splitting by 7H
-- finished splitting by 8H
-- finished splitting by 9H
-- finished splitting by 10H
-- finished splitting by 11H
-- finished splitting by 12H
-- finished splitting by 13H
-- finished splitting by 14H
-- finished splitting by 15H
-- finished splitting by 16H
-- finished splitting by 17H
-- finished splitting by 18H
-- finished splitting by 19H
-- finished splitting by 20H
-- finished splitting by 21H
-- finished splitting by 22H
-- finished splitting by 23H
```

1.5 Feature engineering

In this section, we focus on squeezing out useful information that can be used as features in training and testing our model. The inspiration for our features was consolidating from various readings

This section involves 1. Generating features 2. Reformating dataframes

The data set is aggregated based on unique origin-destination IP addresses. The features that we are using includes: 1. Number of unique origin ports 2. Number of unique destination ports 3. Total number of connections identified 4. Total HTTP Method score 5. Total User-Agent score 6. Total username score 7. Total tag score 8. Total proxy score

```
[ ]: ## read one train set
     # trainset = pd.read_csv('train/train_15.csv', sep='\t', error_bad_lines = False)
     # trainset.ts = pd.to_datetime(trainset.ts)
```

```
[13]: def engineer_datasets(input_fp, output_fp, timebin=5):
          feature_samples = pd.read_csv(input_fp, sep='\t', error_bad_lines = False)
          feature_samples.ts = pd.to_datetime(feature_samples.ts)

          engineered_features_headers = [
              'start_hr', "start_min", 'id.orig_h', 'id.resp_h', "unique_orig_p",␣
        ↪"unique_resp_p", "total_connections",
              "sum_method_score", "sum_ua_score", "sum_username_score",␣
        ↪"sum_tag_score", "sum_proxied_score"]
          output_sample = pd.DataFrame(columns=engineered_features_headers)
          start_hour = input_fp.split("_")[1].split(".")[0]
          print('start hour', start_hour)

          timebin_mins = timebin
          for tbin in range(60 // timebin_mins):
              start_time = tbin*timebin_mins
              d_samples = feature_samples[(start_time <= feature_samples.ts.dt.minute)␣
        ↪& (feature_samples.ts.dt.minute < start_time+4)]
              all_orig_ip = list(d_samples['id.orig_h'].unique())

              ## for all origin ip
              for orig_ip in all_orig_ip:
                  sample_by_orig = d_samples[d_samples['id.orig_h'] == orig_ip]
                  all_resp_ip = list(sample_by_orig['id.resp_h'].unique())

                  ## for all dst ip
                  for resp_ip in all_resp_ip:
                      sample_by_orig_n_dst = sample_by_orig[sample_by_orig['id.
        ↪resp_h'] == resp_ip]

                      unique_orig_p = len(sample_by_orig_n_dst["id.orig_p"].unique())
```

```
                unique_resp_p = len(sample_by_orig_n_dst["id.resp_p"].unique())
                sum_method_score = sample_by_orig_n_dst.method_score.sum()
                sum_ua_score = sample_by_orig_n_dst.ua_score.sum()
                sum_username_score = sample_by_orig_n_dst.username_score.sum()
                sum_tag_score = sample_by_orig_n_dst.tag_score.sum()
                sum_proxied_score = sample_by_orig_n_dst.proxied_score.sum()

                new_data = {
                    "start_hr": start_hour,
                    "start_min": start_time,
                    "id.orig_h": orig_ip,
                    "id.resp_h": resp_ip,
                    "unique_orig_p": unique_orig_p,
                    "unique_resp_p": unique_resp_p,
                    "total_connections": len(sample_by_orig_n_dst),
                    "sum_method_score": sum_method_score,
                    "sum_ua_score": sum_ua_score,
                    "sum_username_score": sum_username_score,
                    "sum_tag_score": sum_tag_score,
                    "sum_proxied_score": sum_proxied_score
                }
                output_sample = output_sample.append(new_data, ignore_index=True)


        print("-- {}-{} timebin complete --".format(start_time, start_time+4))
    output_sample.to_csv(output_fp, sep='\t', index=False)
```

[14]:
```
## enumerate through all files and aggregate data
timebin = 5
for repo in ["train", "test"]:
    for input_file in os.listdir(repo):
        if input_file.endswith(".csv"):
            input_fp = os.path.join(repo, input_file)
            output_fp = os.path.join(repo, "aggregated_" +␣
 ↪str(timebin),input_file)
            if not os.path.exists(os.path.join(repo, "aggregated_" +␣
 ↪str(timebin))):
                os.mkdir(os.path.join(repo, "aggregated_" + str(timebin)))
            engineer_datasets(input_fp, output_fp)
```

```
start hour 12
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 13
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 14
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 15
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\IPython\core\interactiveshell.py:3254: DtypeWarning: Columns (14) have
```

```
mixed types.Specify dtype option on import or set low_memory=False.
  if (await self.run_code(code, result,  async_=asy)):
```

```
start hour 16
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 17
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 18
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 12
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 13
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 14
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 15
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 16
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 17
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 18
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 19
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 20
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 21
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
start hour 22
-- 0-4 timebin complete --
-- 5-9 timebin complete --
-- 10-14 timebin complete --
-- 15-19 timebin complete --
-- 20-24 timebin complete --
-- 25-29 timebin complete --
-- 30-34 timebin complete --
-- 35-39 timebin complete --
-- 40-44 timebin complete --
-- 45-49 timebin complete --
-- 50-54 timebin complete --
-- 55-59 timebin complete --
```

[15]:
```
def combine_datasets(list_of_input_fp, output_fp):
    dfs = []
    for filepath in list_of_input_fp:
        df = pd.read_csv(filepath, sep='\t', error_bad_lines = False)
        dfs.append(df)
    merged_df = pd.concat(dfs, ignore_index=True)
    merged_df.to_csv(output_fp, sep='\t', index=False)
```

[16]:
```
aggregated_directories = ["train\\aggregated_5", "test\\aggregated_5"]
for d in aggregated_directories:
    list_of_datasets = []

    for datasets in os.listdir(d):
        if datasets.startswith("t"):
            input_fp = os.path.join(d, datasets)
            list_of_datasets.append(input_fp)
    output_fp = os.path.join(d, "agg.csv")
    combine_datasets(list_of_datasets, output_fp)
```

[17]:
```
agg_train = pd.read_csv("train\\aggregated_5\\agg.csv", sep='\t',␣
 ↪error_bad_lines = False)
agg_test = pd.read_csv("test\\aggregated_5\\agg.csv", sep='\t', error_bad_lines␣
 ↪= False)
print("Size of new train set with aggregated data: {}".format(agg_train.shape))
print("Size of new test set with aggregated data: {}".format(agg_test.shape))
print("Let's take a look at the first 10 rows of test data\n")
agg_test.head(10)
```

```
Size of new train set with aggregated data: (1217, 12)
Size of new test set with aggregated data: (1522, 12)
Let's take a look at the first 10 rows of test data
```

[17]:
```
   start_hr  start_min        id.orig_h       id.resp_h  unique_orig_p  \
0        12         25   192.168.202.95   192.168.201.2              1
1        12         30  192.168.202.112   192.168.201.2              6
2        12         30   192.168.202.87   192.168.201.2              2
3        12         30   192.168.202.90   192.168.201.2              1
4        12         35   192.168.203.66  192.168.202.78             22
5        12         35  192.168.202.112  192.168.26.253             17
6        12         35  192.168.202.112   192.168.201.2              8
7        12         35   192.168.202.90   192.168.201.2              5
8        12         40   192.168.202.90   192.168.201.2              6
9        12         40  192.168.202.112  192.168.24.253              5

   unique_resp_p  total_connections  sum_method_score  sum_ua_score  \
0              1                  9                 0             0
1              1                 29                 0             0
2              1                  4                 0             0
3              1                  5                 0             0
4              1                 22                 0             0
```

```
5                 1                 106                    0                0
6                 1                  37                    0                0
7                 1                  49                    0                0
8                 1                  67                    0                0
9                 1                   5                    0                0

   sum_username_score  sum_tag_score  sum_proxied_score
0                   0              0                  0
1                   0              0                  0
2                   0              0                  0
3                   0              0                  0
4                   0              0                  0
5                   0              0                  0
6                   0              0                  0
7                   0              0                  0
8                   0              0                  0
9                   0              0                  0
```

1.6 Model Testing

Taking inspiration from the various papers [1][2][3] we identified that Local Outlier Factor (LOF) and One-Class SVM as two possible unsupervised learning algorithms appropriate for our unlabelled dataset situation.

Before running the algorithm, we normalized our data using a simple standard scaler. This normalizes the features into a Gaussian distribution, thus normalizing our scores.

Further improvements

1. PCA or SVD, can be used to compare the importance of the different features or as features into the model. However, since we lose feature context when vectors are decomposed, this should only be considered further into the research.
2. Further tests on other unsupervised models like Isolation Forest, or fine tuning the model parameters can also be used to get the optimal model for this use case.

[18]:
```python
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import LocalOutlierFactor
from sklearn.svm import OneClassSVM
```

[19]:
```python
def results(orig_df, results, flag_valueS):
    flagged_data = pd.DataFrame(columns=list(orig_df.columns.values))

    for idx in range(len(results)):
        if results[idx] in flag_valueS:
            flagged_data = flagged_data.append(orig_df.iloc[idx],␣
↪ignore_index=True)
    print("-- showing results --")
    display(flagged_data)
    return flagged_data
```

[20]:
```
samp = agg_train.drop(['start_hr', "start_min", 'id.orig_h', 'id.resp_h'],␣
 ↪axis=1)
sampT = agg_test.drop(['start_hr', "start_min", 'id.orig_h', 'id.resp_h'],␣
 ↪axis=1)

scale = StandardScaler().fit(samp)
scaled_Base = scale.transform(samp)
scaled_Test = scale.transform(sampT)
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\preprocessing\data.py:625: DataConversionWarning: Data with
input dtype int64 were all converted to float64 by StandardScaler.
  return self.partial_fit(X, y)
C:\Users\kzile\Anaconda3\envs\sml\lib\site-packages\ipykernel_launcher.py:5:
DataConversionWarning: Data with input dtype int64 were all converted to float64
by StandardScaler.
  """
C:\Users\kzile\Anaconda3\envs\sml\lib\site-packages\ipykernel_launcher.py:6:
DataConversionWarning: Data with input dtype int64 were all converted to float64
by StandardScaler.
```

[21]:
```
## Test LOF --> we assume most points are normal (non-recon) data
clf = LocalOutlierFactor(n_neighbors=2, novelty=True)
scores = clf.fit(scaled_Base).predict(scaled_Test)
lof_1 = results(agg_test, scores, [-1])

## Test LOF --> we assume most points are malicious (pentest environment) so we␣
 ↪assume inliers are malicious
clf = LocalOutlierFactor(n_neighbors=10, novelty=True)
scores = clf.fit(scaled_Base).predict(scaled_Test)
lof_2 = results(agg_test, scores, [-1])

## OneClass SVM
model = OneClassSVM(gamma=0.3)
model.fit(scaled_Base)
labels = model.predict(scaled_Test)
ocsvm = results(agg_test, labels, [-1])
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\neighbors\lof.py:236: FutureWarning: default contamination
parameter 0.1 will change in version 0.22 to "auto". This will change the
predict method behavior.
  FutureWarning)
```

```
-- showing results --
```

```
    start_hr start_min        id.orig_h        id.resp_h unique_orig_p  \
0         12        25   192.168.202.95    192.168.201.2             1
1         12        30   192.168.202.87    192.168.201.2             2
2         12        35  192.168.202.112   192.168.26.253            17
3         12        40   192.168.202.90    192.168.201.2             6
4         13        10  192.168.202.103  192.168.229.101             3
..       ...       ...              ...              ...           ...
210       21        45   192.168.204.45   192.168.26.252             7
211       21        45   192.168.204.45   192.168.26.253            18
212       21        45   192.168.204.45   192.168.26.203             4
213       22         5   192.168.204.60   192.168.202.78            10
214       22        30   192.168.202.65    192.168.201.2             1

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1                 9                0            0
1               1                 4                0            0
2               1               106                0            0
3               1                67                0            0
4               1                 4                0            0
..            ...               ...              ...          ...
210             1                 7                0            2
211             2                18                2            7
212             1                 4                0            3
213             1                10                0            0
214             1                20                0            0

    sum_username_score sum_tag_score sum_proxied_score
0                    0             0                 0
1                    0             0                 0
2                    0             0                 0
3                    0             0                 0
4                    0             0                 0
..                 ...           ...               ...
210                  0             0                 0
211                  0             0                 0
212                  0             0                 0
213                  4             0                 0
214                  0             0                 0

[215 rows x 12 columns]
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\neighbors\lof.py:236: FutureWarning: default contamination
parameter 0.1 will change in version 0.22 to "auto". This will change the
predict method behavior.
  FutureWarning)
```

```
-- showing results --
```

```
    start_hr start_min        id.orig_h        id.resp_h unique_orig_p  \
0         12        35   192.168.203.66   192.168.202.78            22
1         12        50  192.168.202.112   192.168.25.253             6
2         12        50  192.168.202.112   192.168.26.253             7
3         13        10  192.168.202.103  192.168.229.101             3
4         13        15  192.168.202.112   192.168.26.253            10
..       ...       ...              ...              ...           ...
172       21        50  192.168.202.110  192.168.229.156            37
173       21        55   192.168.202.87    192.168.201.2             7
174       22         0   192.168.202.76  192.168.229.156            37
175       22        15   192.168.202.76  192.168.229.156            37
176       22        30   192.168.202.76  192.168.229.156            37

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1                22                0            0
1               1                11                0            0
2               1                13                0            0
3               1                 4                0            0
4               1                10                0            0
..            ...               ...              ...          ...
172             1                37                0            0
173             1                14                0            0
174             1                37                0            0
175             1                37                0            0
176             1                37                0            0

    sum_username_score sum_tag_score sum_proxied_score
0                    0             0                 0
1                    0             0                 0
2                    0             0                 0
3                    0             0                 0
4                    0             0                 0
..                 ...           ...               ...
172                 37             0                 0
173                  0             0                 0
174                 37             0                 0
175                 37             0                 0
176                 37             0                 0

[177 rows x 12 columns]


-- showing results --

    start_hr start_min        id.orig_h        id.resp_h unique_orig_p  \
0         12        40  192.168.202.112    192.168.201.2             1
1         12        45   192.168.204.45   192.168.21.253             3
2         12        50   192.168.204.45   192.168.21.253            16
3         13         0   192.168.204.45   192.168.22.253             3
```

```
4         13        5   192.168.204.45   192.168.22.253            15
..       ...      ...              ...              ...           ...
613       22       15   192.168.202.91  192.168.205.253             1
614       22       15   192.168.202.94   192.168.25.252             1
615       22       20   192.168.202.76  192.168.229.156            38
616       22       25   192.168.202.76  192.168.229.156            38
617       22       30   192.168.202.76  192.168.229.156            37

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1                 2                0            0
1               2                 3                2            0
2               2                18                0            7
3               2                 3                2            0
4               2                15                0            7
..            ...               ...              ...          ...
613             1                 1                0            0
614             1                 1                0            0
615             1                38                0            0
616             1                38                0            0
617             1                37                0            0

    sum_username_score sum_tag_score sum_proxied_score
0                    0             0                 0
1                    0             0                 0
2                    0             0                 0
3                    0             0                 0
4                    0             0                 0
..                 ...           ...               ...
613                  0             0                 0
614                  0             0                 0
615                 38             0                 0
616                 38             0                 0
617                 37             0                 0

[618 rows x 12 columns]
```

1.7 Validation

In this section, we will grab the flagged records (outliers are scored -1) from our results above and check it against our original records.

[22]:
```
lof_1["id.orig_h"].value_counts()
```

[22]:
```
192.168.202.65                           22
192.168.204.45                           19
192.168.202.138                          17
192.168.202.112                          15
```

```
192.168.202.110                           13
192.168.202.79                            13
192.168.202.140                           13
192.168.202.90                             9
192.168.202.4                              9
192.168.202.103                            8
192.168.202.87                             8
192.168.203.45                             6
192.168.202.144                            5
192.168.203.64                             4
192.168.202.63                             4
192.168.202.108                            4
192.168.203.63                             4
192.168.202.125                            3
192.168.202.95                             3
192.168.202.109                            3
192.168.202.136                            3
192.168.204.70                             3
192.168.202.122                            3
192.168.202.102                            3
192.168.202.143                            3
192.168.202.62                             2
192.168.202.94                             2
2001:dbb:c18:202:20c:29ff:fe41:4be7        2
192.168.202.118                            2
192.168.202.152                            1
192.168.202.88                             1
192.168.202.100                            1
2001:dbb:c18:202:20c:29ff:fe93:571e        1
192.168.202.150                            1
192.168.202.141                            1
192.168.202.64                             1
192.168.202.76                             1
192.168.204.60                             1
192.168.202.115                            1
Name: id.orig_h, dtype: int64
```

```
[23]: lof_2["id.orig_h"].value_counts()
```

```
[23]: 192.168.202.140                           23
      192.168.202.4                             19
      192.168.202.110                           19
      192.168.202.76                            16
      2001:dbb:c18:202:20c:29ff:fe93:571e       12
      192.168.202.79                            10
      192.168.202.103                            7
      192.168.202.138                            7
```

```
192.168.202.112                           6
192.168.202.144                           6
192.168.202.65                            4
192.168.204.70                            4
192.168.204.45                            4
192.168.203.64                            4
192.168.202.122                           3
192.168.202.136                           3
192.168.202.143                           3
192.168.202.64                            3
192.168.202.153                           2
192.168.202.90                            2
192.168.202.94                            2
192.168.202.87                            2
2001:dbb:c18:202:20c:29ff:fe41:4be7       2
192.168.203.45                            2
192.168.202.152                           1
192.168.202.68                            1
192.168.202.88                            1
192.168.203.66                            1
192.168.202.141                           1
192.168.202.62                            1
192.168.202.125                           1
192.168.202.109                           1
192.168.202.102                           1
192.168.202.100                           1
192.168.202.63                            1
192.168.202.108                           1
Name: id.orig_h, dtype: int64
```

[24]:
```
ocsvm["id.orig_h"].value_counts()
```

[24]:
```
192.168.202.140                         104
192.168.202.79                           64
192.168.202.110                          61
192.168.202.76                           46
192.168.202.112                          45
192.168.202.4                            41
192.168.202.108                          38
192.168.202.102                          27
192.168.202.138                          25
192.168.204.45                           20
192.168.202.136                          13
2001:dbb:c18:202:20c:29ff:fe93:571e      11
192.168.202.94                            8
192.168.26.100                            8
192.168.202.101                           7
```

```
192.168.202.144                        7
192.168.203.45                         7
192.168.204.70                         6
192.168.203.63                         6
192.168.202.103                        6
192.168.203.64                         5
192.168.202.143                        5
192.168.202.90                         5
2001:dbb:c18:202:20c:29ff:fe18:b667    5
192.168.202.122                        4
192.168.202.125                        4
192.168.27.100                         4
2001:dbb:c18:202:20c:29ff:fe41:4be7    4
192.168.202.87                         3
192.168.202.91                         3
192.168.202.109                        2
192.168.202.141                        2
192.168.202.88                         2
192.168.202.222                        2
192.168.202.118                        2
192.168.202.96                         1
192.168.202.153                        1
192.168.24.253                         1
192.168.202.98                         1
192.168.204.60                         1
192.168.202.95                         1
192.168.202.135                        1
192.168.22.253                         1
192.168.28.100                         1
192.168.28.253                         1
192.168.202.100                        1
192.168.202.115                        1
192.168.202.62                         1
192.168.202.150                        1
192.168.202.68                         1
192.168.21.253                         1
Name: id.orig_h, dtype: int64
```

[25]:
```
agg_test[agg_test["id.orig_h"] == "192.168.202.140"]
```

[25]:
```
      start_hr  start_min        id.orig_h       id.resp_h  unique_orig_p  \
132         14         55  192.168.202.140  192.168.21.103              2
133         14         55  192.168.202.140  192.168.21.102              1
134         14         55  192.168.202.140  192.168.21.253              3
135         14         55  192.168.202.140  192.168.22.253              3
136         14         55  192.168.202.140  192.168.23.253              1
...        ...        ...              ...             ...            ...
```

```
1076        19          45  192.168.202.140  192.168.25.103             24
1116        19          50  192.168.202.140  192.168.25.103             16
1150        19          55  192.168.202.140  192.168.25.103              1
1162        20           0  192.168.202.140  192.168.25.103             12
1193        20           5  192.168.202.140  192.168.25.103              3

      unique_resp_p  total_connections  sum_method_score  sum_ua_score  \
132               2                  2                 0             0
133               1                  1                 0             0
134               2                  3                 2             0
135               2                  3                 2             0
136               1                  1                 0             0
...             ...                ...               ...           ...
1076              1                110                 0             0
1116              1                 34                 0             0
1150              1                  6                 0             0
1162              1                 27                 0             0
1193              1                  3                 0             0

      sum_username_score  sum_tag_score  sum_proxied_score
132                    0              0                  0
133                    0              0                  0
134                    0              0                  0
135                    0              0                  0
136                    0              0                  0
...                  ...            ...                ...
1076                   0              0                  0
1116                   0              0                  0
1150                   0              0                  0
1162                   0              0                  0
1193                   0              0                  0

[183 rows x 12 columns]
```

1.8 Improving model

We observe from the previous section that the results are not accurate, flagged data IP addresses contain a mix of *observably* normal and malicious data points. To further improve results, we remove as many malicious data points as possible from the training dataset. This should give a more accurate model for normal (inlier) data points.

1. Run model.fit_predict on training dataset for all models. This returns outliers within the training dataset. We assume this to be malicious/reconnaissance effort.
2. Move these data from train to testing dataset
3. Rerun tests

[26]:
```
## Test LOF --> we assume most points are normal (non-recon) data
clf = LocalOutlierFactor(n_neighbors=2)
scores = clf.fit_predict(scaled_Base)
improv_lof_1 = results(agg_test, scores, [-1])

## Test LOF --> we assume most points are malicious (pentest environment) so we␣
 ↪assume inliers are malicious
clf = LocalOutlierFactor(n_neighbors=10)
scores = clf.fit_predict(scaled_Base)
improv_lof_2 = results(agg_test, scores, [-1])

## OneClass SVM
model = OneClassSVM(gamma=0.3)
labels = model.fit_predict(scaled_Base)
# labels = model.predict(scaled_Test)
improv_ocsvm = results(agg_test, labels, [-1])
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\neighbors\lof.py:236: FutureWarning: default contamination
parameter 0.1 will change in version 0.22 to "auto". This will change the
predict method behavior.
  FutureWarning)
```

```
-- showing results --

    start_hr start_min          id.orig_h        id.resp_h unique_orig_p  \
0         12        25    192.168.202.95    192.168.201.2             1
1         12        35    192.168.202.90    192.168.201.2             5
2         12        40   192.168.202.112    192.168.201.2             1
3         12        50    192.168.204.45   192.168.202.78             4
4         13        10   192.168.202.103  192.168.229.101             3
..       ...       ...               ...              ...           ...
117       20         5     192.168.202.4   192.168.26.103             1
118       20         5   192.168.202.102   192.168.23.253             1
119       20         5   192.168.202.102   192.168.23.152             1
120       20        10    192.168.28.100   192.168.202.82             2
121       20        15    192.168.202.65    192.168.201.2             1

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1                 9                0            0
1               1                49                0            0
2               1                 2                0            0
3               1                 4                0            0
4               1                 4                0            0
..            ...               ...              ...          ...
117             1                 1                0            0
118             1                 1                0            0
119             1                 1                0            0
```

```
120             1                2               0            0
121             1               48               0            0

    sum_username_score sum_tag_score sum_proxied_score
0                    0             0                 0
1                    0             0                 0
2                    0             0                 0
3                    0             0                 0
4                    0             0                 0
..                 ...           ...               ...
117                  0             0                 0
118                  0             0                 0
119                  1             0                 0
120                  0             0                 0
121                  0             0                 0

[122 rows x 12 columns]
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\neighbors\lof.py:236: FutureWarning: default contamination
parameter 0.1 will change in version 0.22 to "auto". This will change the
predict method behavior.
  FutureWarning)
```

```
-- showing results --
```

```
    start_hr start_min        id.orig_h        id.resp_h unique_orig_p  \
0         12        30  192.168.202.112    192.168.201.2             6
1         12        30   192.168.202.87    192.168.201.2             2
2         12        30   192.168.202.90    192.168.201.2             1
3         13        10  192.168.202.103  192.168.229.101             3
4         13        15  192.168.202.112   192.168.26.253            10
..       ...       ...              ...              ...           ...
116       20         5    192.168.202.4   192.168.26.152             1
117       20         5    192.168.202.4   192.168.23.103             1
118       20         5   192.168.28.100   192.168.202.82             2
119       20         5  192.168.202.141   192.168.23.102            53
120       20        10  192.168.202.103   192.168.25.202            15

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1                29                0            0
1               1                 4                0            0
2               1                 5                0            0
3               1                 4                0            0
4               1                10                0            0
..            ...               ...              ...          ...
116             1                 1                0            0
117             1                 1                0            0
```

```
118             1                2               0            0
119             1               53               0            0
120             1               16               0            0

    sum_username_score sum_tag_score sum_proxied_score
0                    0             0                 0
1                    0             0                 0
2                    0             0                 0
3                    0             0                 0
4                    0             0                 0
..                 ...           ...               ...
116                  0             0                 0
117                  0             0                 0
118                  0             0                 0
119                  0             0                 0
120                  0             0                 0

[121 rows x 12 columns]


-- showing results --
    start_hr start_min         id.orig_h       id.resp_h unique_orig_p  \
0         12        25    192.168.202.95   192.168.201.2             1
1         12        35   192.168.202.112  192.168.26.253            17
2         12        35   192.168.202.112   192.168.201.2             8
3         12        35    192.168.202.90   192.168.201.2             5
4         12        40    192.168.202.90   192.168.201.2             6
..       ...       ...               ...             ...           ...
604       20        15   192.168.202.103  192.168.25.202            16
605       20        15    192.168.202.65   192.168.201.2             1
606       20        15    192.168.204.45  192.168.21.253             4
607       20        15    192.168.202.79  192.168.24.203             3
608       20        15    192.168.202.79  192.168.28.103             1

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1                 9                0            0
1               1               106                0            0
2               1                37                0            0
3               1                49                0            0
4               1                67                0            0
..            ...               ...              ...          ...
604             1                16                0            0
605             1                48                0            0
606             1                 4                0            0
607             2                 9                0            0
608             1                 1                0            0

    sum_username_score sum_tag_score sum_proxied_score
```

```
0                    0             0                  0
1                    0             0                  0
2                    0             0                  0
3                    0             0                  0
4                    0             0                  0
..                 ...           ...                ...
604                  0             0                  0
605                  0             0                  0
606                  0             0                  0
607                  4             0                  0
608                  0             0                  0

[609 rows x 12 columns]
```

[27]:
```
improv_train = agg_train
improv_test = agg_test

print("Moving the following origin IP addresses...")
for df in [improv_lof_1, improv_lof_2, improv_ocsvm]:
    for ip in list(df["id.orig_h"].value_counts()[:5].index):
        print(ip)
        improv_test = improv_test.append(improv_train[improv_train["id.orig_h"]␣
 ↪== ip], ignore_index=True)
        improv_train = improv_train.drop(improv_train[improv_train["id.orig_h"]␣
 ↪== ip].index)
```

```
Moving the following origin IP addresses...
192.168.202.140
192.168.202.138
192.168.202.4
192.168.202.112
192.168.202.79
192.168.202.79
192.168.202.112
192.168.202.140
192.168.202.4
192.168.202.65
192.168.202.140
192.168.202.112
192.168.202.79
192.168.202.103
192.168.202.65
```

[28]:
```
improv_samp = improv_train.drop(['start_hr', "start_min", 'id.orig_h', 'id.
 ↪resp_h'], axis=1)
improv_sampT = improv_test.drop(['start_hr', "start_min", 'id.orig_h', 'id.
 ↪resp_h'], axis=1)
```

```
scale = StandardScaler().fit(improv_samp)
scaled_Base = scale.transform(improv_samp)
scaled_Test = scale.transform(improv_sampT)
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\preprocessing\data.py:625: DataConversionWarning: Data with
input dtype int64 were all converted to float64 by StandardScaler.
  return self.partial_fit(X, y)
C:\Users\kzile\Anaconda3\envs\sml\lib\site-packages\ipykernel_launcher.py:5:
DataConversionWarning: Data with input dtype int64 were all converted to float64
by StandardScaler.
  """
C:\Users\kzile\Anaconda3\envs\sml\lib\site-packages\ipykernel_launcher.py:6:
DataConversionWarning: Data with input dtype int64 were all converted to float64
by StandardScaler.
```

[29]:
```
## Test LOF --> we assume most points are normal (non-recon) data
clf = LocalOutlierFactor(n_neighbors=2, novelty=True)
scores = clf.fit(scaled_Base).predict(scaled_Test)
f_lof_1 = results(improv_test, scores, [-1])

## Test LOF --> we assume most points are malicious (pentest environment) so we␣
 ↪assume inliers are malicious
clf = LocalOutlierFactor(n_neighbors=10, novelty=True)
scores = clf.fit(scaled_Base).predict(scaled_Test)
f_lof_2 = results(improv_test, scores, [-1])

## OneClass SVM
model = OneClassSVM(gamma=0.3)
model.fit(scaled_Base)
labels = model.predict(scaled_Test)
f_ocsvm = results(improv_test, labels, [-1])
```

```
C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\neighbors\lof.py:236: FutureWarning: default contamination
parameter 0.1 will change in version 0.22 to "auto". This will change the
predict method behavior.
  FutureWarning)

-- showing results --

   start_hr start_min        id.orig_h        id.resp_h unique_orig_p  \
0        12        35  192.168.202.112   192.168.26.253            17
1        12        50  192.168.202.112   192.168.25.253             6
2        12        50  192.168.202.112   192.168.26.253             7
3        12        50   192.168.204.45   192.168.21.253            16
```

```
4         13         5   192.168.204.45   192.168.22.253            15
..       ...       ...              ...              ...           ...
304       18        15   192.168.202.79  192.168.229.251            15
305       17        30  192.168.202.103   192.168.25.202            14
306       17        30  192.168.202.103   192.168.23.202             5
307       17        45  192.168.202.103   192.168.22.202             9
308       18         0  192.168.202.103   192.168.24.101             4

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1               106                0            0
1               1                11                0            0
2               1                13                0            0
3               2                18                0            7
4               2                15                0            7
..            ...               ...              ...          ...
304             1                15                0           14
305             1                16                0            0
306             1                19                0            0
307             1                11                0            0
308             1                 7                0            0

    sum_username_score sum_tag_score sum_proxied_score
0                    0             0                 0
1                    0             0                 0
2                    0             0                 0
3                    0             0                 0
4                    0             0                 0
..                 ...           ...               ...
304                  0             0                 0
305                  0             0                 0
306                  0             0                 0
307                  0             0                 0
308                  0             0                 0

[309 rows x 12 columns]


C:\Users\kzile\Anaconda3\envs\sml\lib\site-
packages\sklearn\neighbors\lof.py:236: FutureWarning: default contamination
parameter 0.1 will change in version 0.22 to "auto". This will change the
predict method behavior.
  FutureWarning)

-- showing results --

    start_hr start_min        id.orig_h        id.resp_h unique_orig_p  \
0         12        35   192.168.203.66   192.168.202.78            22
1         12        50  192.168.202.112   192.168.25.253             6
2         12        50  192.168.202.112   192.168.26.253             7
```

```
3          12         50   192.168.204.45  192.168.202.119              6
4          13         10  192.168.202.103  192.168.229.101              3
..        ...        ...              ...              ...            ...
289        18         15   192.168.202.79  192.168.229.251             15
290        16         30  192.168.202.103   192.168.23.202              9
291        17         10  192.168.202.103   192.168.25.202             13
292        17         25  192.168.202.103  192.168.229.101              2
293        17         45  192.168.202.103   192.168.22.202              9

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1                22                0            0
1               1                11                0            0
2               1                13                0            0
3               1                15                0            0
4               1                 4                0            0
..            ...               ...              ...          ...
289             1                15                0           14
290             1                25                0            0
291             1                13                0            0
292             1                 5                0            0
293             1                11                0            0

    sum_username_score sum_tag_score sum_proxied_score
0                    0             0                 0
1                    0             0                 0
2                    0             0                 0
3                    0             0                 0
4                    0             0                 0
..                 ...           ...               ...
289                  0             0                 0
290                  0             0                 0
291                  0             0                 0
292                  0             0                 0
293                  0             0                 0

[294 rows x 12 columns]


-- showing results --

    start_hr start_min        id.orig_h        id.resp_h unique_orig_p  \
0          12         40  192.168.202.112    192.168.201.2              1
1          12         45   192.168.204.45   192.168.21.253              3
2          12         50   192.168.204.45   192.168.21.253             16
3          13          0   192.168.204.45   192.168.22.253              3
4          13          5   192.168.204.45   192.168.22.253             15
..        ...        ...              ...              ...            ...
770        18         15   192.168.202.79  192.168.229.153              5
771        18         20   192.168.202.79  192.168.229.101              1
```

```
772        13         30  192.168.202.103  192.168.229.156           21
773        15         55  192.168.202.103   192.168.24.202            1
774        18         10  192.168.202.103   192.168.24.101            1

    unique_resp_p total_connections sum_method_score sum_ua_score  \
0               1                 2                0            0
1               2                 3                2            0
2               2                18                0            7
3               2                 3                2            0
4               2                15                0            7
..            ...               ...              ...          ...
770             1                 5                0            4
771             1                 2                0            0
772             1                21                0            0
773             1                 1                0            0
774             1                 1                0            0

    sum_username_score sum_tag_score sum_proxied_score
0                    0             0                 0
1                    0             0                 0
2                    0             0                 0
3                    0             0                 0
4                    0             0                 0
..                 ...           ...               ...
770                  0             0                 0
771                  0             0                 0
772                  8             0                 0
773                  0             0                 0
774                  0             0                 0

[775 rows x 12 columns]
```

[30]:
```
f_lof_1["id.orig_h"].value_counts()
```

[30]:
```
192.168.202.79                     49
192.168.202.140                    35
192.168.202.112                    28
192.168.202.103                    20
192.168.202.138                    19
192.168.202.65                     18
192.168.202.4                      18
192.168.204.45                     15
2001:dbb:c18:202:20c:29ff:fe93:571e    14
192.168.202.110                    14
192.168.202.87                      8
192.168.202.144                     8
192.168.202.90                      8
```

```
192.168.204.70                      5
192.168.202.63                      4
192.168.203.63                      4
192.168.202.108                     4
192.168.202.143                     3
192.168.202.102                     3
192.168.202.125                     3
192.168.202.109                     3
192.168.202.94                      3
192.168.203.64                      3
192.168.202.141                     3
192.168.202.118                     2
192.168.202.136                     2
2001:dbb:c18:202:20c:29ff:fe41:4be7     2
192.168.203.45                      2
192.168.202.76                      2
192.168.23.254                      1
192.168.202.95                      1
192.168.202.150                     1
192.168.202.152                     1
192.168.202.100                     1
192.168.202.88                      1
192.168.204.60                      1
Name: id.orig_h, dtype: int64
```

[31]:
```
f_lof_2["id.orig_h"].value_counts()
```

[31]:
```
192.168.202.140                    57
192.168.202.79                     38
192.168.202.4                      30
192.168.202.112                    19
192.168.202.110                    19
2001:dbb:c18:202:20c:29ff:fe93:571e    18
192.168.202.76                     16
192.168.204.45                     14
192.168.202.138                    10
192.168.202.103                     8
192.168.202.122                     7
192.168.202.144                     6
192.168.202.64                      6
192.168.202.136                     5
192.168.203.64                      5
192.168.202.65                      4
192.168.202.143                     4
192.168.203.45                      3
192.168.202.90                      3
192.168.202.109                     2
```

```
192.168.202.62                       2
192.168.202.87                       2
192.168.202.68                       2
192.168.202.141                      2
192.168.202.153                      2
192.168.202.94                       2
192.168.202.152                      1
192.168.203.66                       1
192.168.204.70                       1
192.168.202.102                      1
192.168.202.100                      1
192.168.202.108                      1
192.168.202.63                       1
192.168.202.88                       1
Name: id.orig_h, dtype: int64
```

[32]:
```
f_ocsvm["id.orig_h"].value_counts()
```

[32]:
```
192.168.202.140                    147
192.168.202.79                     140
192.168.202.4                       63
192.168.202.110                     60
192.168.202.76                      46
192.168.202.108                     38
192.168.202.112                     35
192.168.204.45                      33
2001:dbb:c18:202:20c:29ff:fe93:571e     31
192.168.202.102                     27
192.168.202.138                     23
192.168.202.136                     13
192.168.202.103                      9
192.168.26.100                       8
192.168.202.94                       8
192.168.202.144                      7
192.168.202.101                      7
192.168.203.63                       6
192.168.203.45                       6
192.168.204.70                       6
192.168.202.143                      5
192.168.203.64                       5
2001:dbb:c18:202:20c:29ff:fe18:b667      5
192.168.202.90                       4
2001:dbb:c18:202:20c:29ff:fe41:4be7      4
192.168.202.125                      4
192.168.27.100                       4
192.168.202.91                       3
192.168.202.122                      3
```

```
192.168.202.87                        2
192.168.202.118                       2
192.168.202.141                       2
192.168.202.88                        2
192.168.22.253                        1
192.168.202.150                       1
192.168.202.222                       1
192.168.202.153                       1
192.168.202.68                        1
192.168.204.60                        1
192.168.202.98                        1
192.168.24.253                        1
192.168.28.253                        1
192.168.28.100                        1
192.168.202.135                       1
192.168.202.115                       1
192.168.202.95                        1
192.168.202.100                       1
192.168.202.109                       1
192.168.202.96                        1
192.168.21.253                        1
Name: id.orig_h, dtype: int64
```

[33]:
```
test[test["id.orig_h"] == "192.168.202.79"].head(50)
```

[33]:
```
                                   ts                 uid       id.orig_h  \
1600640 2012-03-16 19:33:48.509999989   CUgXz91gXLm5ukPbJ  192.168.202.79
1600641 2012-03-16 19:33:48.660000086   CUgXz91gXLm5ukPbJ  192.168.202.79
1600642 2012-03-16 19:33:48.710000038   CUgXz91gXLm5ukPbJ  192.168.202.79
1600645 2012-03-16 19:33:53.210000038   CUgXz91gXLm5ukPbJ  192.168.202.79
1600646 2012-03-16 19:33:53.440000057   CUgXz91gXLm5ukPbJ  192.168.202.79
1600647 2012-03-16 19:33:58.339999914  CPQ5Sv1buAHcrm2Y95  192.168.202.79
1600648 2012-03-16 19:33:58.339999914  CZGImS2hOGhUnk6bK8  192.168.202.79
1600649 2012-03-16 19:33:58.339999914  Cn1OWF2PrsFBYlEfce  192.168.202.79
1600650 2012-03-16 19:33:58.339999914  C3p6yJ3nc6lxm9HzGl  192.168.202.79
1600651 2012-03-16 19:33:58.289999962   CUgXz91gXLm5ukPbJ  192.168.202.79
1600652 2012-03-16 19:33:59.009999990  C3p6yJ3nc6lxm9HzGl  192.168.202.79
1600655 2012-03-16 19:34:06.259999990  C3p6yJ3nc6lxm9HzGl  192.168.202.79
1600656 2012-03-16 19:34:06.630000114  C3p6yJ3nc6lxm9HzGl  192.168.202.79
1600657 2012-03-16 19:34:07.069999933  C3p6yJ3nc6lxm9HzGl  192.168.202.79
1600658 2012-03-16 19:34:07.140000105  C3p6yJ3nc6lxm9HzGl  192.168.202.79
1600660 2012-03-16 19:34:13.849999905  C3p6yJ3nc6lxm9HzGl  192.168.202.79
1600676 2012-03-16 19:34:21.519999981  C3p6yJ3nc6lxm9HzGl  192.168.202.79
1600721 2012-03-16 19:34:50.740000010   CzNrCkx0eE9vEQVc3  192.168.202.79
1600724 2012-03-16 19:34:51.130000114   CzNrCkx0eE9vEQVc3  192.168.202.79
1600725 2012-03-16 19:34:51.130000114   CzNrCkx0eE9vEQVc3  192.168.202.79
1600726 2012-03-16 19:34:51.130000114   CzNrCkx0eE9vEQVc3  192.168.202.79
```

```
1600727 2012-03-16 19:34:51.150000095   CzNrCkxOeE9vEQVc3  192.168.202.79
1600728 2012-03-16 19:34:51.150000095   CzNrCkxOeE9vEQVc3  192.168.202.79
1600729 2012-03-16 19:34:51.160000086   CzNrCkxOeE9vEQVc3  192.168.202.79
1600730 2012-03-16 19:34:51.160000086   CzNrCkxOeE9vEQVc3  192.168.202.79
1600731 2012-03-16 19:34:51.170000076   CzNrCkxOeE9vEQVc3  192.168.202.79
1600732 2012-03-16 19:34:51.170000076   CzNrCkxOeE9vEQVc3  192.168.202.79
1600733 2012-03-16 19:34:51.170000076   CzNrCkxOeE9vEQVc3  192.168.202.79
1600734 2012-03-16 19:34:51.170000076   CzNrCkxOeE9vEQVc3  192.168.202.79
1600735 2012-03-16 19:34:51.170000076   CzNrCkxOeE9vEQVc3  192.168.202.79
1600736 2012-03-16 19:34:51.180000067   CzNrCkxOeE9vEQVc3  192.168.202.79
1600737 2012-03-16 19:34:51.180000067   CzNrCkxOeE9vEQVc3  192.168.202.79
1600738 2012-03-16 19:34:51.180000067   CzNrCkxOeE9vEQVc3  192.168.202.79
1600739 2012-03-16 19:34:51.180000067   CzNrCkxOeE9vEQVc3  192.168.202.79
1600740 2012-03-16 19:34:51.180000067   CzNrCkxOeE9vEQVc3  192.168.202.79
1600741 2012-03-16 19:34:51.180000067   CzNrCkxOeE9vEQVc3  192.168.202.79
1600742 2012-03-16 19:34:51.190000057   CzNrCkxOeE9vEQVc3  192.168.202.79
1600743 2012-03-16 19:34:51.190000057   CzNrCkxOeE9vEQVc3  192.168.202.79
1600744 2012-03-16 19:34:51.190000057   CzNrCkxOeE9vEQVc3  192.168.202.79
1600745 2012-03-16 19:34:51.190000057   CzNrCkxOeE9vEQVc3  192.168.202.79
1600746 2012-03-16 19:34:51.190000057   CzNrCkxOeE9vEQVc3  192.168.202.79
1600747 2012-03-16 19:34:51.190000057   CzNrCkxOeE9vEQVc3  192.168.202.79
1600748 2012-03-16 19:34:51.200000048   CzNrCkxOeE9vEQVc3  192.168.202.79
1600749 2012-03-16 19:34:51.200000048   CzNrCkxOeE9vEQVc3  192.168.202.79
1600750 2012-03-16 19:34:51.200000048   CzNrCkxOeE9vEQVc3  192.168.202.79
1600751 2012-03-16 19:34:51.200000048   CzNrCkxOeE9vEQVc3  192.168.202.79
1600752 2012-03-16 19:34:51.200000048   CzNrCkxOeE9vEQVc3  192.168.202.79
1600753 2012-03-16 19:34:51.210000038   CzNrCkxOeE9vEQVc3  192.168.202.79
1600754 2012-03-16 19:34:51.210000038   CzNrCkxOeE9vEQVc3  192.168.202.79
1600755 2012-03-16 19:34:51.230000019   CzNrCkxOeE9vEQVc3  192.168.202.79

         id.orig_p       id.resp_h  id.resp_p  trans_depth method  \
1600640      48761  192.168.25.203         80            1    GET
1600641      48761  192.168.25.203         80            2    GET
1600642      48761  192.168.25.203         80            3    GET
1600645      48761  192.168.25.203         80            4    GET
1600646      48761  192.168.25.203         80            5    GET
1600647      48765  192.168.25.203         80            1    GET
1600648      48763  192.168.25.203         80            1    GET
1600649      48764  192.168.25.203         80            1    GET
1600650      48766  192.168.25.203         80            1    GET
1600651      48761  192.168.25.203         80            6    GET
1600652      48766  192.168.25.203         80            2    GET
1600655      48766  192.168.25.203         80            3   POST
1600656      48766  192.168.25.203         80            4    GET
1600657      48766  192.168.25.203         80            5    GET
1600658      48766  192.168.25.203         80            6    GET
1600660      48766  192.168.25.203         80            7    GET
```

```
1600676      48766  192.168.25.203        80           8    GET
1600721      48769  192.168.25.203        80           1   HEAD
1600724      48769  192.168.25.203        80           2    GET
1600725      48769  192.168.25.203        80           3    GET
1600726      48769  192.168.25.203        80           4    GET
1600727      48769  192.168.25.203        80           5    GET
1600728      48769  192.168.25.203        80           6    GET
1600729      48769  192.168.25.203        80           7    GET
1600730      48769  192.168.25.203        80           8    GET
1600731      48769  192.168.25.203        80           9    GET
1600732      48769  192.168.25.203        80          10    GET
1600733      48769  192.168.25.203        80          11    GET
1600734      48769  192.168.25.203        80          12    GET
1600735      48769  192.168.25.203        80          13    GET
1600736      48769  192.168.25.203        80          14    GET
1600737      48769  192.168.25.203        80          15    GET
1600738      48769  192.168.25.203        80          16    GET
1600739      48769  192.168.25.203        80          17    GET
1600740      48769  192.168.25.203        80          18    GET
1600741      48769  192.168.25.203        80          19    GET
1600742      48769  192.168.25.203        80          20    GET
1600743      48769  192.168.25.203        80          21    GET
1600744      48769  192.168.25.203        80          22    GET
1600745      48769  192.168.25.203        80          23    GET
1600746      48769  192.168.25.203        80          24    GET
1600747      48769  192.168.25.203        80          25    GET
1600748      48769  192.168.25.203        80          26    GET
1600749      48769  192.168.25.203        80          27    GET
1600750      48769  192.168.25.203        80          28    GET
1600751      48769  192.168.25.203        80          29    GET
1600752      48769  192.168.25.203        80          30    GET
1600753      48769  192.168.25.203        80          31    GET
1600754      48769  192.168.25.203        80          32    GET
1600755      48769  192.168.25.203        80          33    GET

                   host                                               uri  \
1600640  192.168.25.203                                                 /
1600641  192.168.25.203                                      /favicon.ico
1600642  192.168.25.203                                      /favicon.ico
1600645  192.168.25.203                                       /phpmyadmin
1600646  192.168.25.203                                      /phpmyadmin/
1600647  192.168.25.203        /phpmyadmin/themes/original/img/b_help.png
1600648  192.168.25.203                             /phpmyadmin/print.css
1600649  192.168.25.203    /phpmyadmin/themes/original/img/logo_right.png
1600650  192.168.25.203                           /phpmyadmin/favicon.ico
1600651  192.168.25.203  /phpmyadmin/phpmyadmin.css.php?lang=en-utf-8&c...
1600652  192.168.25.203      /phpmyadmin/themes/original/img/s_notice.png
```

```
1600655  192.168.25.203                          /phpmyadmin/index.php
1600656  192.168.25.203  /phpmyadmin/index.php?token=f2aa4efebcc1d0a2cb...
1600657  192.168.25.203  /phpmyadmin/phpmyadmin.css.php?token=f2aa4efeb...
1600658  192.168.25.203        /phpmyadmin/themes/original/img/s_error.png
1600660  192.168.25.203                                           /openemr
1600676  192.168.25.203                                               /oer
1600721  192.168.25.203                                                  /
1600724  192.168.25.203                                                  /
1600725  192.168.25.203                                                  /
1600726  192.168.25.203                                    /aoY7kzbH.html+
1600727  192.168.25.203                                    /aoY7kzbH.php3+
1600728  192.168.25.203                                      /aoY7kzbH.fhp
1600729  192.168.25.203                                     /aoY7kzbH.stat
1600730  192.168.25.203                                     /aoY7kzbH.conf
1600731  192.168.25.203                                      /aoY7kzbH.nlm
1600732  192.168.25.203                                /aoY7kzbH.00RelNotes
1600733  192.168.25.203                                      /aoY7kzbH.cnf
1600734  192.168.25.203                                      /aoY7kzbH.tcl
1600735  192.168.25.203                                      /aoY7kzbH.dat
1600736  192.168.25.203                                      /aoY7kzbH.dll
1600737  192.168.25.203                                      /aoY7kzbH.mdb
1600738  192.168.25.203                                       /aoY7kzbH.es
1600739  192.168.25.203                                   /aoY7kzbH.iso-ru
1600740  192.168.25.203                            /aoY7kzbH.VALIDATE_STMT
1600741  192.168.25.203                            /aoY7kzbH.BBoardServlet
1600742  192.168.25.203                                   /aoY7kzbH.js0x70
1600743  192.168.25.203                                    /aoY7kzbH.thtml
1600744  192.168.25.203                                      /aoY7kzbH.txt
1600745  192.168.25.203                                      /aoY7kzbH.cfm
1600746  192.168.25.203                                        /aoY7kzbH.c
1600747  192.168.25.203                                      /aoY7kzbH.org
1600748  192.168.25.203                                      /aoY7kzbH.nsf
1600749  192.168.25.203                                   /aoY7kzbH.config
1600750  192.168.25.203                                      /aoY7kzbH.exe
1600751  192.168.25.203                                      /aoY7kzbH.gif
1600752  192.168.25.203                                     /aoY7kzbH.shtm
1600753  192.168.25.203                                    /aoY7kzbH.pt-br
1600754  192.168.25.203                                     /aoY7kzbH.aspx
1600755  192.168.25.203                                      /aoY7kzbH.INC

                                              referrer  \
1600640                                              -
1600641                                              -
1600642                                              -
1600645                                              -
1600646                                              -
1600647                 http://192.168.25.203/phpmyadmin/
```

```
1600648                 http://192.168.25.203/phpmyadmin/
1600649                 http://192.168.25.203/phpmyadmin/
1600650                                                 -
1600651                 http://192.168.25.203/phpmyadmin/
1600652  http://192.168.25.203/phpmyadmin/phpmyadmin.cs...
1600655                 http://192.168.25.203/phpmyadmin/
1600656                 http://192.168.25.203/phpmyadmin/
1600657  http://192.168.25.203/phpmyadmin/index.php?tok...
1600658  http://192.168.25.203/phpmyadmin/phpmyadmin.cs...
1600660                                                 -
1600676                                                 -
1600721                                                 -
1600724                                                 -
1600725                                                 -
1600726                                                 -
1600727                                                 -
1600728                                                 -
1600729                                                 -
1600730                                                 -
1600731                                                 -
1600732                                                 -
1600733                                                 -
1600734                                                 -
1600735                                                 -
1600736                                                 -
1600737                                                 -
1600738                                                 -
1600739                                                 -
1600740                                                 -
1600741                                                 -
1600742                                                 -
1600743                                                 -
1600744                                                 -
1600745                                                 -
1600746                                                 -
1600747                                                 -
1600748                                                 -
1600749                                                 -
1600750                                                 -
1600751                                                 -
1600752                                                 -
1600753                                                 -
1600754                                                 -
1600755                                                 -

                                               user_agent  request_body_len  \
1600640  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...                 0
```

```
1600641  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600642  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600645  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600646  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600647  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600648  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600649  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600650  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600651  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600652  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600655  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...          287
1600656  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600657  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600658  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600660  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600676  Mozilla/5.0 (X11; Linux i686; rv:10.0.2) Gecko...            0
1600721  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600724  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600725  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600726  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600727  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600728  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600729  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600730  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600731  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600732  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600733  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600734  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600735  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600736  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600737  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600738  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600739  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600740  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600741  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600742  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600743  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600744  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600745  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600746  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600747  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600748  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600749  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600750  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600751  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600752  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600753  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...            0
1600754  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...                0
1600755  Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Te...                0

         response_body_len status_code         status_msg info_code info_msg  \
1600640                177         200                 OK         -        -
1600641                289         404          Not Found         -        -
1600642                289         404          Not Found         -        -
1600645                321         301  Moved Permanently         -        -
1600646               8625         200                 OK         -        -
1600647                138         200                 OK         -        -
1600648               1063         200                 OK         -        -
1600649               4756         200                 OK         -        -
1600650              18902         200                 OK         -        -
1600651              21786         200                 OK         -        -
1600652                145         200                 OK         -        -
1600655                  0         302              Found         -        -
1600656               7633         200                 OK         -        -
1600657              21786         200                 OK         -        -
1600658                162         200                 OK         -        -
1600660                285         404          Not Found         -        -
1600676                281         404          Not Found         -        -
1600721                  0         200                 OK         -        -
1600724                177         200                 OK         -        -
1600725                177         200                 OK         -        -
1600726                292         404          Not Found         -        -
1600727                292         404          Not Found         -        -
1600728                290         404          Not Found         -        -
1600729                291         404          Not Found         -        -
1600730                291         404          Not Found         -        -
1600731                290         404          Not Found         -        -
1600732                297         404          Not Found         -        -
1600733                290         404          Not Found         -        -
1600734                290         404          Not Found         -        -
1600735                290         404          Not Found         -        -
1600736                290         404          Not Found         -        -
1600737                290         404          Not Found         -        -
1600738                289         404          Not Found         -        -
1600739                293         404          Not Found         -        -
1600740                300         404          Not Found         -        -
1600741                300         404          Not Found         -        -
1600742                293         404          Not Found         -        -
1600743                292         404          Not Found         -        -
1600744                290         404          Not Found         -        -
1600745                290         404          Not Found         -        -
1600746                288         404          Not Found         -        -
1600747                290         404          Not Found         -        -
1600748                290         404          Not Found         -        -
```

```
1600749                293          404         Not Found          -        -
1600750                290          404         Not Found          -        -
1600751                290          404         Not Found          -        -
1600752                291          404         Not Found          -        -
1600753                292          404         Not Found          -        -
1600754                291          404         Not Found          -        -
1600755                290          404         Not Found          -        -

        filename     tags username password proxied          orig_fuids  \
1600640        -  (empty)        -        -       -                   -
1600641        -  (empty)        -        -       -                   -
1600642        -  (empty)        -        -       -                   -
1600645        -  (empty)        -        -       -                   -
1600646        -  (empty)        -        -       -                   -
1600647        -  (empty)        -        -       -                   -
1600648        -  (empty)        -        -       -                   -
1600649        -  (empty)        -        -       -                   -
1600650        -  (empty)        -        -       -                   -
1600651        -  (empty)        -        -       -                   -
1600652        -  (empty)        -        -       -                   -
1600655        -  (empty)        -        -       -  FApwh5492cIaXnm6ae
1600656        -  (empty)        -        -       -                   -
1600657        -  (empty)        -        -       -                   -
1600658        -  (empty)        -        -       -                   -
1600660        -  (empty)        -        -       -                   -
1600676        -  (empty)        -        -       -                   -
1600721        -  (empty)        -        -       -                   -
1600724        -  (empty)        -        -       -                   -
1600725        -  (empty)        -        -       -                   -
1600726        -  (empty)        -        -       -                   -
1600727        -  (empty)        -        -       -                   -
1600728        -  (empty)        -        -       -                   -
1600729        -  (empty)        -        -       -                   -
1600730        -  (empty)        -        -       -                   -
1600731        -  (empty)        -        -       -                   -
1600732        -  (empty)        -        -       -                   -
1600733        -  (empty)        -        -       -                   -
1600734        -  (empty)        -        -       -                   -
1600735        -  (empty)        -        -       -                   -
1600736        -  (empty)        -        -       -                   -
1600737        -  (empty)        -        -       -                   -
1600738        -  (empty)        -        -       -                   -
1600739        -  (empty)        -        -       -                   -
1600740        -  (empty)        -        -       -                   -
1600741        -  (empty)        -        -       -                   -
1600742        -  (empty)        -        -       -                   -
1600743        -  (empty)        -        -       -                   -
```

```
1600744        -  (empty)          -        -       -                   -
1600745        -  (empty)          -        -       -                   -
1600746        -  (empty)          -        -       -                   -
1600747        -  (empty)          -        -       -                   -
1600748        -  (empty)          -        -       -                   -
1600749        -  (empty)          -        -       -                   -
1600750        -  (empty)          -        -       -                   -
1600751        -  (empty)          -        -       -                   -
1600752        -  (empty)          -        -       -                   -
1600753        -  (empty)          -        -       -                   -
1600754        -  (empty)          -        -       -                   -
1600755        -  (empty)          -        -       -                   -

        orig_mine_types          resp_fuids resp_mime_types  method_score  \
1600640               -  FlEaYY3DfTo4vkK0aa       text/html             0
1600641               -  Freh1v1hbyKsEm6bpk       text/html             0
1600642               -  FPYHPj41bayxM8BWX8       text/html             0
1600645               -  F3uDil1zb3YvhfFtsb       text/html             0
1600646               -   FwfCTw4RNk9d7NncF       text/html             0
1600647               -  FlKuN634ud3lK1yyH1       image/png             0
1600648               -  FWPhPd1EKoRV4VA2Mk      text/plain             0
1600649               -  Fb4nv11gx0FRT9Jol9       image/png             0
1600650               -  FoAb511gyoeqhXQyW4    image/x-icon             0
1600651               -  FLTKBq4O9rXqiU6h5a      text/plain             0
1600652               -  FUGW7z4a2JxhWJAogl       image/png             0
1600655      text/plain                   -               -             0
1600656               -   F1loLGSfxE0ah0eb4       text/html             0
1600657               -  FwnRKv4KqfqCZTZ5jb      text/plain             0
1600658               -  FUkGxi3H3jLixlZFg3       image/png             0
1600660               -  FOVWh81xdzFiJ3hjHf       text/html             0
1600676               -  FR3gZ8375Wcor4ZECc       text/html             0
1600721               -                   -               -             0
1600724               -  FwFRsC3GCb4lKLOts4       text/html             0
1600725               -  FMvayA1GECbVY1es15       text/html             0
1600726               -  FzRvpR3GZbEyQ253g6       text/html             0
1600727               -  Fgkj442m4QIWAoaeVk       text/html             0
1600728               -   FI4G9u5TNrUAtVzWe       text/html             0
1600729               -  FOKPkF47anrtUsP3Kc       text/html             0
1600730               -   Fxem0OAyDZT5mIycg       text/html             0
1600731               -  FQU8Fh2vDPBVIbc9Ql       text/html             0
1600732               -  FolV5x15eYvsCCwgYb       text/html             0
1600733               -   FdWMcv4Kj0LYRnd38       text/html             0
1600734               -  FTIQ7w42q9Yh05fA39       text/html             0
1600735               -  FThEky2B2RpUs94e1k       text/html             0
1600736               -  FFwUh72aR1w1DxZ6Ul       text/html             0
1600737               -  FTL9gZ2nD2kz8m6mAl       text/html             0
1600738               -  FsEk1t37oHEqf0mctl       text/html             0
```

```
1600739               -   FF9jsxPWGfClHVJQ8       text/html             0
1600740               -  FCxsan2Ia6GGOlBYll       text/html             0
1600741               -  FhRi494o3Wm1YFeMKc       text/html             0
1600742               -   Fi1sSA2IJ0hTuBMyD       text/html             0
1600743               -  FCf9PO2a6tAmeeRCc6       text/html             0
1600744               -   FDJYu3TlSqi2NbaXk       text/html             0
1600745               -  FLPuZI2vm4Fx75XeIb       text/html             0
1600746               -  FC0pJb4AiCeRP1PaV9       text/html             0
1600747               -   FtKry9hGLYW9K60V1       text/html             0
1600748               -  FMbhNF2cugXkiAbRHg       text/html             0
1600749               -   F1omCUoye6Du12lB3       text/html             0
1600750               -  FvGpe725994n4ydun1       text/html             0
1600751               -   FMK94MuSPUpiAq3Md       text/html             0
1600752               -  FMoVcE1jYmH8uoBpW2       text/html             0
1600753               -  F6h2v32APWvyvIA8c8       text/html             0
1600754               -  FM7sr52kBhy8k1OHkf       text/html             0
1600755               -  FsiE1r2iXPHwbrsRp5       text/html             0

         ua_score  username_score  tag_score  proxied_score
1600640         0               0          0              0
1600641         0               0          0              0
1600642         0               0          0              0
1600645         0               0          0              0
1600646         0               0          0              0
1600647         0               0          0              0
1600648         0               0          0              0
1600649         0               0          0              0
1600650         0               0          0              0
1600651         0               0          0              0
1600652         0               0          0              0
1600655         0               0          0              0
1600656         0               0          0              0
1600657         0               0          0              0
1600658         0               0          0              0
1600660         0               0          0              0
1600676         0               0          0              0
1600721         1               0          0              0
1600724         1               0          0              0
1600725         1               0          0              0
1600726         1               0          0              0
1600727         1               0          0              0
1600728         1               0          0              0
1600729         1               0          0              0
1600730         1               0          0              0
1600731         1               0          0              0
1600732         1               0          0              0
1600733         1               0          0              0
```

```
1600734           1                 0            0                0
1600735           1                 0            0                0
1600736           1                 0            0                0
1600737           1                 0            0                0
1600738           1                 0            0                0
1600739           1                 0            0                0
1600740           1                 0            0                0
1600741           1                 0            0                0
1600742           1                 0            0                0
1600743           1                 0            0                0
1600744           1                 0            0                0
1600745           1                 0            0                0
1600746           1                 0            0                0
1600747           1                 0            0                0
1600748           1                 0            0                0
1600749           1                 0            0                0
1600750           1                 0            0                0
1600751           1                 0            0                0
1600752           1                 0            0                0
1600753           1                 0            0                0
1600754           1                 0            0                0
1600755           1                 0            0                0
```

1.9 Evaluation

From the above results of flagged IP addresses, we take the top 5 IP addresses that were ranked highest across the three models as running network reconnaissance:

1. 192.168.202.140
2. 192.168.202.79
3. 192.168.202.4
4. 192.168.202.110

5. 192.168.202.112

With more experimentation or more records, we can come up with a threshold in which an IP address is flagged only when it occurs more than X number of times within the test set.

Naturally, all IP addresses in the above four cells are flagged as performing reconnaissance, but results confidence is higher for the top 5 commin IP addresses. A list of all flagged IP addresses is below.

[46]:
```python
from collections import Counter
collated_list = Counter()
all_outputs = [f_ocsvm["id.orig_h"].value_counts(), f_lof_2["id.orig_h"].
 ↪value_counts(), f_lof_1["id.orig_h"].value_counts()]
for model in all_outputs:
    for ip in model.keys():
        collated_list[ip] += model[ip]
```

```python
print("Top 5 most common IP address flagged")
for top_ip in collated_list.most_common(5):
    print("IP: {}\tNumber of flagged occurences: {}".format(top_ip[0],␣
 ↪top_ip[1]))
print("==========================================================")
print("All IP addresses flagged")
for ips in collated_list.keys():
    print(ips)
```

```
Top 5 most common IP address flagged
IP: 192.168.202.140     Number of flagged occurences: 239
IP: 192.168.202.79      Number of flagged occurences: 227
IP: 192.168.202.4       Number of flagged occurences: 111
IP: 192.168.202.110     Number of flagged occurences: 93
IP: 192.168.202.112     Number of flagged occurences: 82
==========================================================
All IP addresses flagged
192.168.202.140
192.168.202.79
192.168.202.4
192.168.202.110
192.168.202.76
192.168.202.108
192.168.202.112
192.168.204.45
2001:dbb:c18:202:20c:29ff:fe93:571e
192.168.202.102
192.168.202.138
192.168.202.136
192.168.202.103
192.168.26.100
192.168.202.94
192.168.202.144
192.168.202.101
192.168.203.63
192.168.203.45
192.168.204.70
192.168.202.143
192.168.203.64
2001:dbb:c18:202:20c:29ff:fe18:b667
192.168.202.90
2001:dbb:c18:202:20c:29ff:fe41:4be7
192.168.202.125
192.168.27.100
192.168.202.91
192.168.202.122
192.168.202.87
```

```
192.168.202.118
192.168.202.141
192.168.202.88
192.168.22.253
192.168.202.150
192.168.202.222
192.168.202.153
192.168.202.68
192.168.204.60
192.168.202.98
192.168.24.253
192.168.28.253
192.168.28.100
192.168.202.135
192.168.202.115
192.168.202.95
192.168.202.100
192.168.202.109
192.168.202.96
192.168.21.253
192.168.202.64
192.168.202.65
192.168.202.62
192.168.202.152
192.168.203.66
192.168.202.63
192.168.23.254
```

1.10 Sources

[1] Iain, D (2017, July) Text Classification of Network Intrusion Alerts to Enhance Cyber Situation Awareness and Automate Alert Triage. Department of Defence, Australian Government. Retrieved from: https://www.dst.defence.gov.au/sites/default/files/publications/documents/DST-Group-TN-1640.pdf

[2] Millican, A. (2003, January) Network Reconnaissance - Detection and Prevention. SANS Institute. Retrieved from: https://www.giac.org/paper/gsec/2473/network-reconnaissance-detection-prevention/104296

[3] Swapneel, M. (2018, Decemeber) Anomaly Detection for Network Connection Logs. Arxiv, Cornell University. Retrieved from https://arxiv.org/ftp/arxiv/papers/1812/1812.01941.pdf

[4] Dataset source https://www.secrepo.com/Datasets%20Description/Network/http.html

[5] Ingham, K. (2006, September) Learning DFA representations of HTTP for protecting web applications. Elsevier. Retrieved from: https://crypto.stanford.edu/portia/papers/paper.pdf