Explanatory Analysis on Anomalous Network Traffic

Trisha Chakraborty

April 22, 2021

Motivation:

A purpose of this study is to understand the working mechanism of network intrusion detection or IP blocking algorithms. Network Intrusion Detection Systems are described as a black-box mainly governed by Machine Learning algorithms. This study is a attempt to learn feature extraction of complex network packet data, and to learn, observe and reason the importance of one feature vector over the other.

Dataset:

Anomalous network traffic detection is a complex process given the vast range of attacks that exist, considering the right dataset to focus on the kind of intrusion detection you want to perform. There exists two broad catagory of dataset for detection and evaluation of network traffic:

(1) Flow-based Data Flow based data are meta data of network connection. This data is mostly from the perspective of the host/responder/server. Flow-based datasets provides a summary of connection between host A to host B, maybe unidirectional or bidirectional.

(2) Packet-based Data Packet-based Data are straightforward sniffed packet information from the network. These are bare-bone PCAP files containing the packet information between host A to host B. Disadvantage of using packet based data is a long preprocessing job to convert packet-based to flow-based data for ease of explanatory analysis, which I feel is out of the scope of this final project. Initially I considered using CSS2017- which is a PCAP file format. The dataset I used to perform this project falls in the last catagory.

(3) Other Data This project is performed using KDD-Cup 1999 dataset [3] which falls in other data catagory. This catagory contains some add-on data in the dataset to help perform the analysis.

"A good example of the difference between captured packet inspection and NetFlow would be viewing a forest by hiking through the forest as opposed to flying over the forest in a hot air balloon" [4]

Some important features of the dataset:

1. Raw PCAP files processed to CSV.
2. Unique 145, 586 columns.
3. Column information corresponds to data flow information from client to host.
4. 42 features.
5. Other data.

```
data<-read.table("C:\\Users\\Trisha\\Downloads\\kddcup.data_10_percent_corrected.csv",sep = ",")
```

```
dim(data)
```

```
## [1] 494021     42
```

Feature vectors and column names:

As per the KDD-CUP 1999 dataset[3], underlying are the column names as named in the task descriptions. There are 42 original columns, each representing feature of the network packet.

```
colnames(data) = c("duration", "protocol_type", "service", "flag", "src_bytes", "dst_bytes",
                   "land", "wrong_fragment", "urgent", "hot", "num_failed_logins", "logged_in",
                   "num_compromised", "root_shell", "su_attempted", "num_root", "num_file_creations",
                   "num_shells", "num_access_files", "num_outbound_cmds", "is_host_login",
                   "is_guest_login", "count", "srv_count", "serror_rate", "srv_serror_rate",
                   "rerror_rate","srv_rerror_rate", "same_srv_rate", "diff_srv_rate",
                   "srv_diff_host_rate","dst_host_count","dst_host_srv_count",
                   "dst_host_same_srv_rate", "dst_host_diff_srv_rate",
                   "dst_host_same_src_port_rate", "dst_host_srv_diff_host_rate",
                   "dst_host_serror_rate","dst_host_srv_serror_rate", "dst_host_rerror_rate",
                   "dst_host_srv_rerror_rate", "result")
```

Attack types:

The dataset consists of data for four broad category of network attacks. These attacks are further divided into 24 sub-categorical attack.

(1) DOS: denial-of-service, e.g. syn flood;
(2) R2L: unauthorized access from a remote machine, e.g. guessing password;
(3) U2R: unauthorized access to local superuser (root) privileges, e.g., various “buffer overflow’ ’ attacks;
(4) Probing: surveillance and other probing, e.g., port scanning.

```
data$attack_type[data$result == "ipsweep."] = "probe"
data$attack_type[data$result == "portsweep."] = "probe"
data$attack_type[data$result == "nmap."] = "probe"
data$attack_type[data$result == "satan."] = "probe"
data$attack_type[data$result == "buffer_overflow."] = "u2r"
data$attack_type[data$result == "loadmodule."] = "u2r"
data$attack_type[data$result == "perl."] = "u2r"
data$attack_type[data$result == "rootkit."] = "u2r"
data$attack_type[data$result == "back."] = "dos"
data$attack_type[data$result == "land."] = "dos"
data$attack_type[data$result == "neptune."] = "dos"
data$attack_type[data$result == "pod."] = "dos"
data$attack_type[data$result == "smurf."] = "dos"
data$attack_type[data$result == "teardrop."] = "dos"
data$attack_type[data$result == "ftp_write."] = "r2l"
data$attack_type[data$result == "guess_passwd."] = "r2l"
data$attack_type[data$result == "imap."] = "r2l"
data$attack_type[data$result == "multihop."] = "r2l"
data$attack_type[data$result == "phf."] = "r2l"
data$attack_type[data$result == "spy."] = "r2l"
data$attack_type[data$result == "warezclient."] = "r2l"
data$attack_type[data$result == "warezmaster."] = "r2l"
data$attack_type[data$result == "normal."] = "normal"
```

Data Preprocessing

Data preprocessing was a fairly easy task for this dataset. There were hardly any NA values, but a majority of the columns were duplicated. Removed the duplicate rows from the dataset.

```
sum(duplicated(data))
```

```
## [1] 348435
```

```
dim(data)
```

```
## [1] 494021     43
```

```
data <- data[ !duplicated(data), ]
dim(data)
```

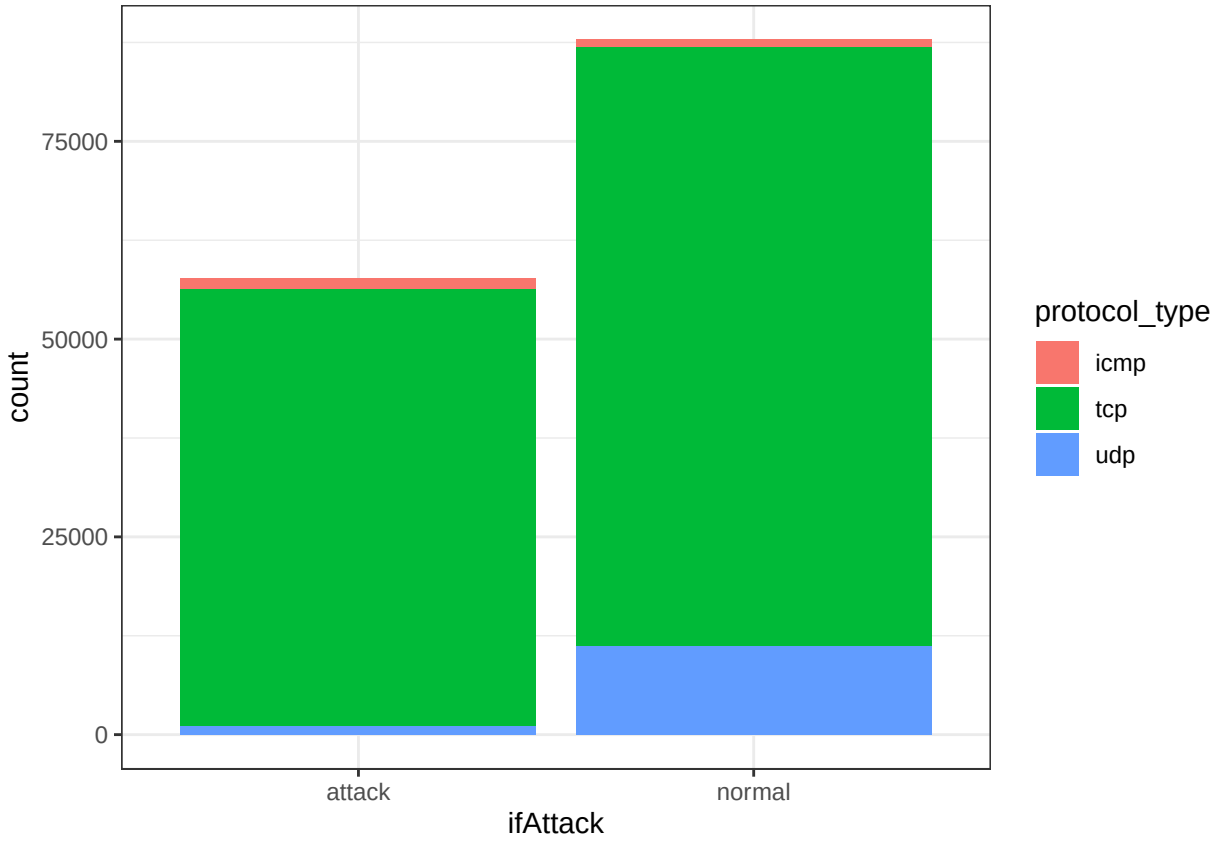
```
## [1] 145586     43
```

Dataset composition - Protocol

Added a column for differentiate between malicious and legitimate traffic.

```
data$ifAttack[data$result != "normal."] = "attack"
data$ifAttack[data$result == "normal."] = "normal"
```

```
ggplot(data) +
  geom_bar(mapping = aes(x = ifAttack, fill = protocol_type)) + theme_bw()
```



Observation:

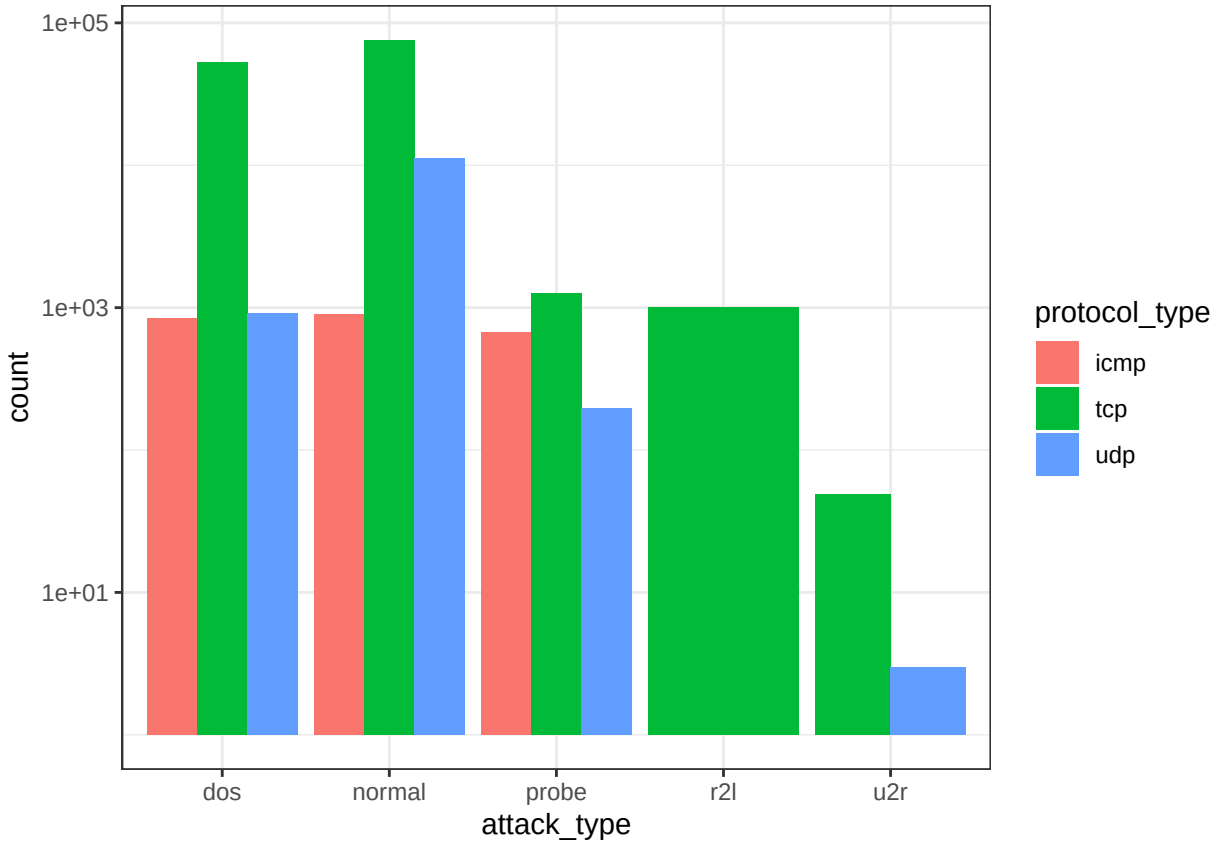
- The dataset has majority legitimate traffic.

From this point, I am interested to observe and reason the malicious records (~60,000 records). Several research has been performed [cite] risk based analysis to identify the feature vectors that directly or indirectly hint at the network traffic being malicious.

Dataset Composition - Type of Attack

The dataset consist of 145, 586 columns after preprocessing. The preprocessing process includes (1) removing N.A. values and duplicate rows, (2) Adding column names as provided by UCI dataset. Tidy-ing the data is an important part of data visualization and most importantly to form observation on such data.

```
ggplot(data) +
  geom_bar(mapping = aes(x = attack_type, fill = protocol_type),position="dodge")+scale_y_log10()+theme_
```

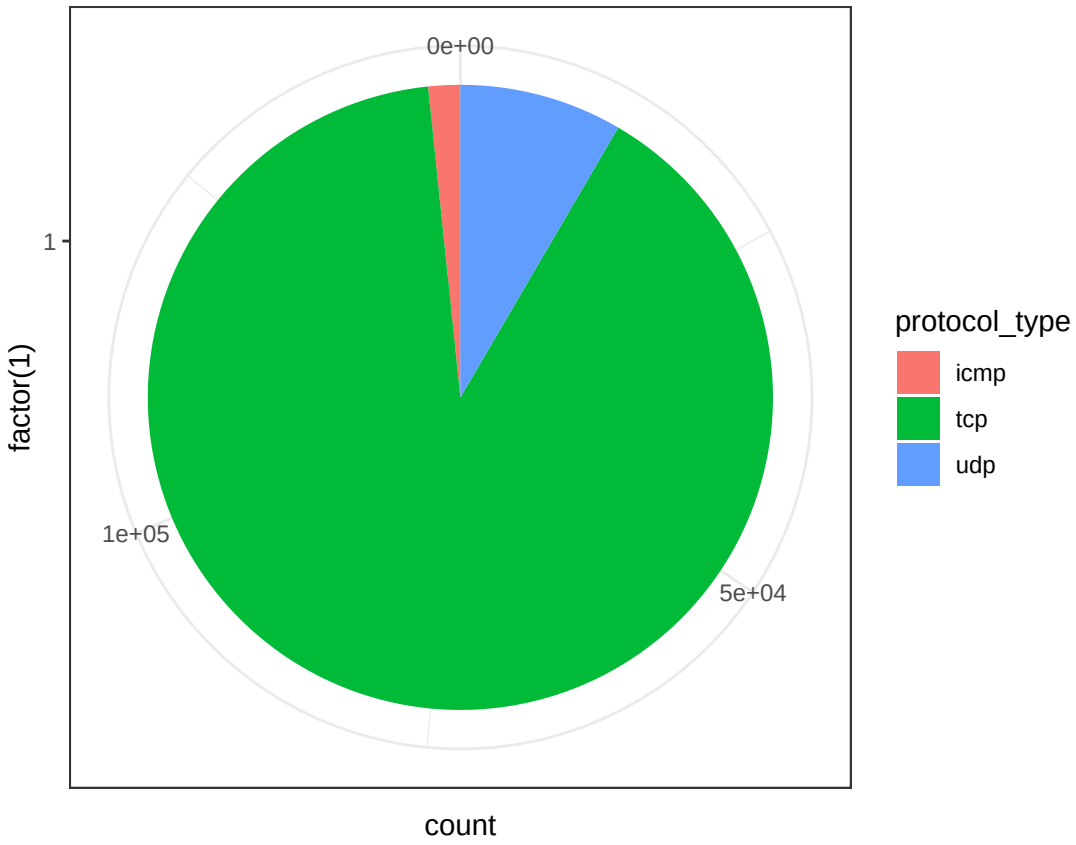


Evidently, normal traffic forms the majority of the data. Among attack data, DOS forms the majority of attack information.

Observation:

- TCP forms the most used protocol type for attack data.

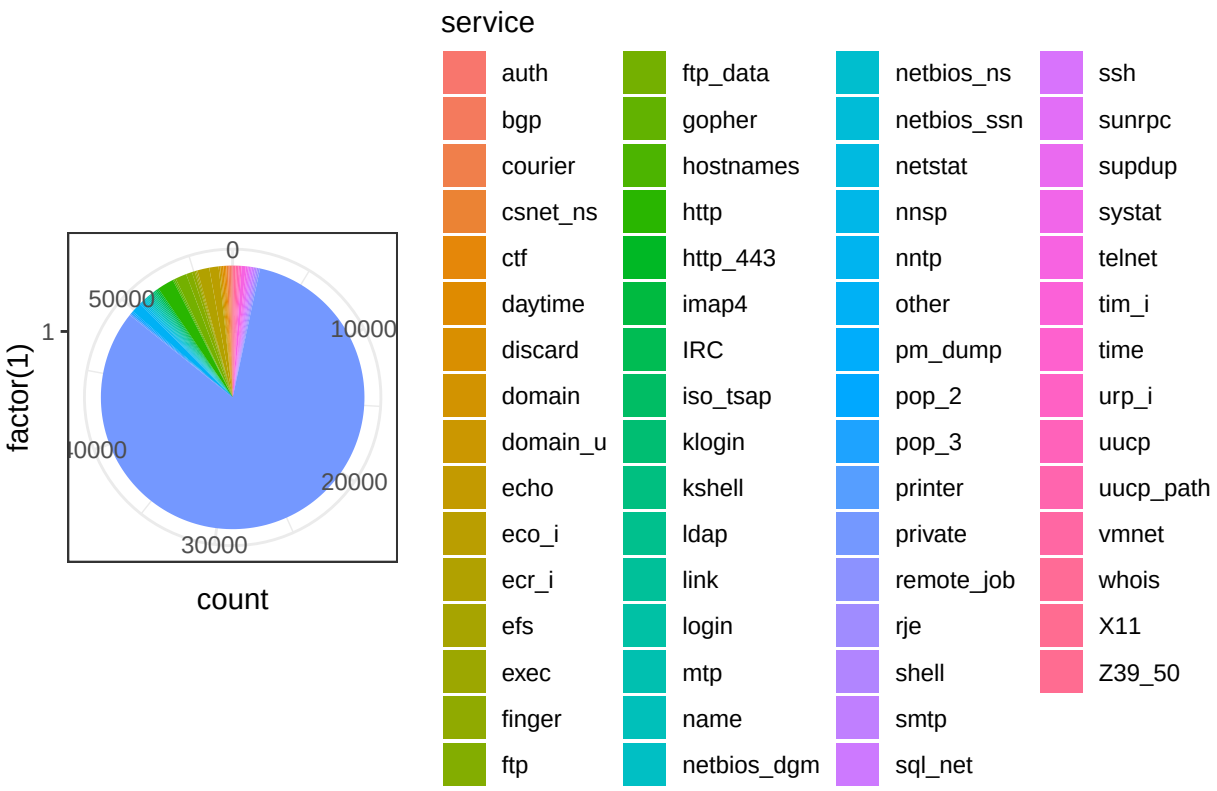
```
ggplot(data, aes(x = factor(1), fill = protocol_type)) +
  geom_bar(width = 1) +
  coord_polar(theta = "y")+theme_bw()
```



Observation: + The dataset has normal majority traffic.

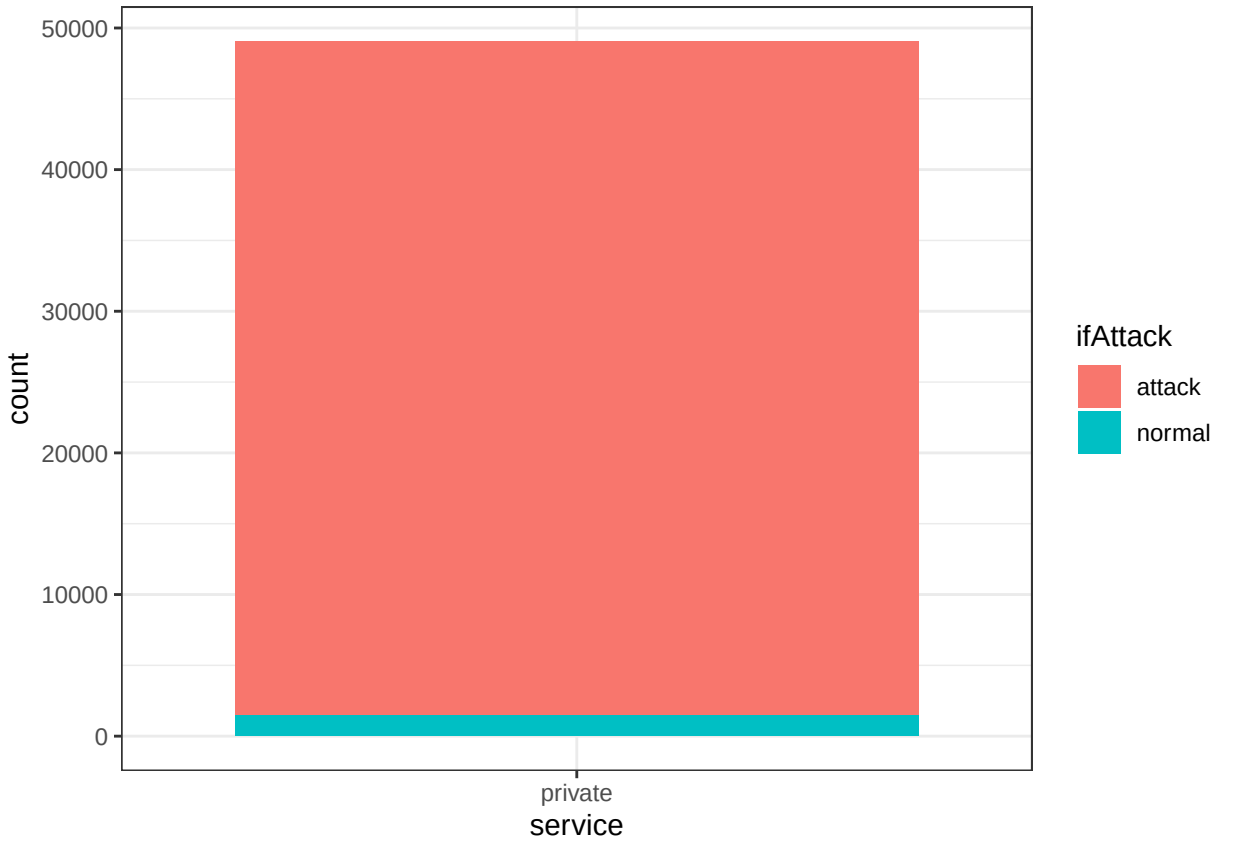
DOS attack has the most traffic in 4 other kind of attack. The protocol composition in TCP mostly. Investigating the type of service used.

```
filter(data, ifAttack == "attack") %>% ggplot(aes(x = factor(1), fill = service)) +
  geom_bar(width = 1) +
  coord_polar(theta = "y")+theme_bw()
```



The service used to perform DOS was mostly private. But is private only part of attack traffic?

```
filter(data, service == "private") %>% ggplot() +
  geom_bar(mapping = aes(x = service, fill = ifAttack))+theme_bw()
```

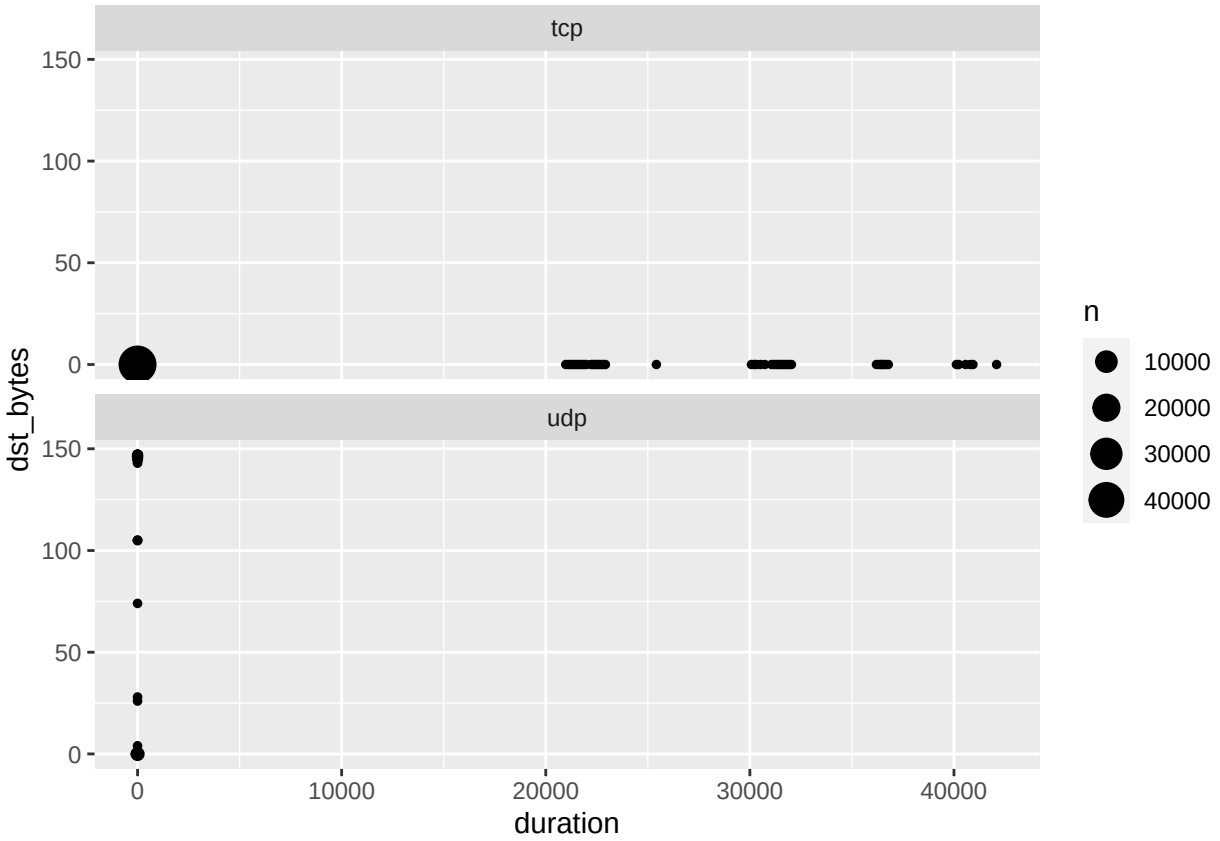


Observation: Some service = private are part of normal traffic, probably users private IP for using VPN.

```
filter(data,service == "private" & ifAttack =="normal")
```

Precisely, 1,463 records found for users with private service. Investigating about what possible reason?

```
filter(data, service == "private") %>% ggplot(mapping = aes(x = duration, y = dst_bytes)) +
  geom_count()+
  facet_wrap(~ protocol_type, nrow = 2)
```



The normal traffic with private protocol mostly uses TCP. This traffic doesn't seem to be false negative because of the following observation: 1. UDP traffic is short lived in the network, hence the duration is close to 0. UDP traffic gets acceptable response from the destination. 2. A large proportion of TCP traffic is short-lived. Only few has longer duration of traffic.

Explanatory Data Analysis

KDD-Cup 1999 data consists of 42 feature vectors, with 7 symbolic feature vectors and remaining continuous data. The symbolic feature vectors are extra information about the flow of the packet collected from the host side on top of network packet information - this makes KDD-Cup 1999 data different from flow based network traffic information. The symbolic feature vectors are [1]:

(1) Land : If connection is from the same host. Possible value 0 or 1.
(2) Logged In : 1 if successfully logged in; 0 otherwise.
(3) Host Login Data: 1 if the login belongs to the "hot' ' list; 0 otherwise.
(4) Guest Login Data: 1 if the login is a "guest' 'login; 0 otherwise.
(5) Flag: Normal or error status of the connection.

Apart from the above symbolic feature vector which has been explored earlier by various researchers, I explored the following feature vectors and performed explanatory data analysis to learn the importance of these feature (at all) in determining legitimacy of network packet. These are:

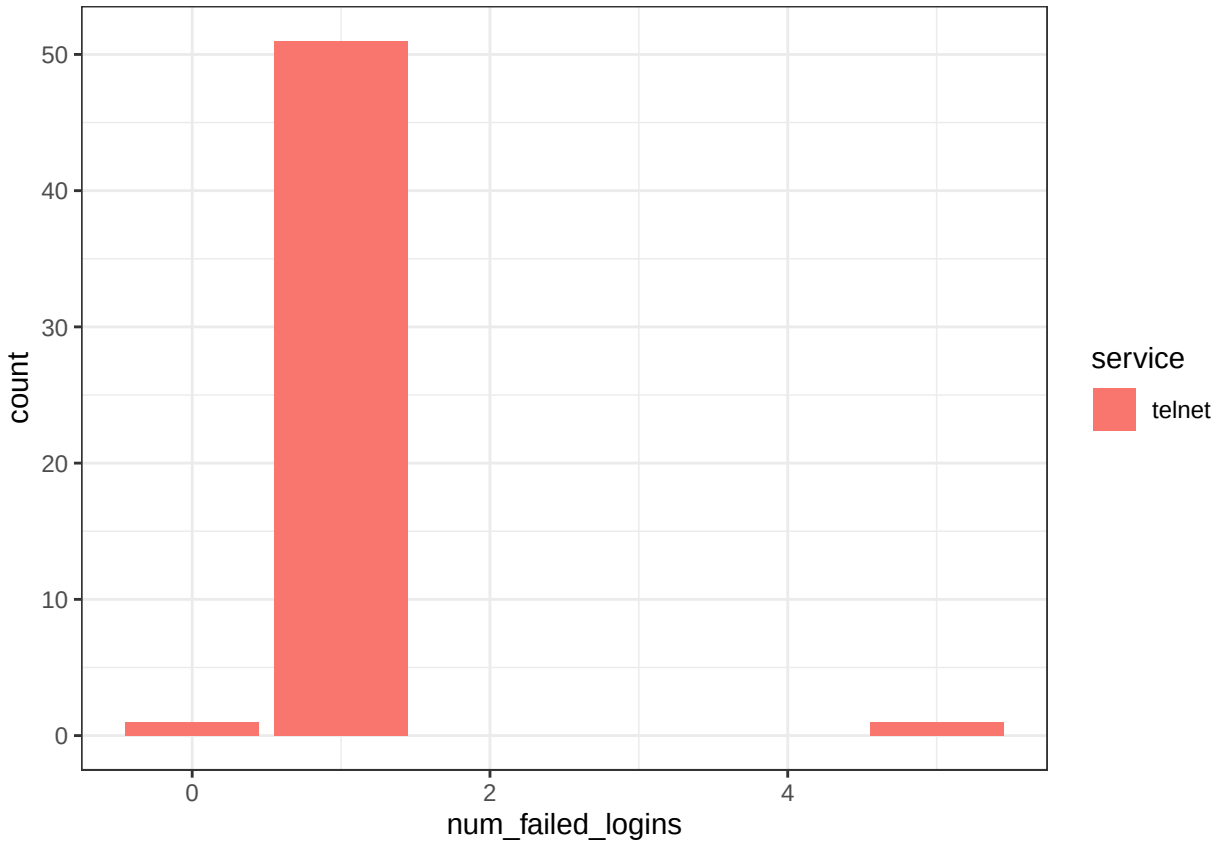
(1) Number of failed attempt indicator
(2) Port blasting indicator
(3) Flag indicator
(4) Duration
(5) Source data bytes indicator
(6) Number of compromised indicator

(7) Guest login indicator

Number of failed login attempt indicator

Number of failed login attempt is the count of the number of times a login attempt has failed. To analyze this indicator, let's focus on Guess Password attack. Guess Password is a probe attack where attacker bruteforces password of legitimate user.

```
filter(data, result == "guess_passwd.") %>% ggplot ()+
  geom_bar(mapping = aes(x = num_failed_logins, fill = service))+theme_bw()
```



Observation:

- Telnet is the only service used for guess password attack.
- Guess password has majority num_failed_logins = 1.

The fact that Telnet service is the entirety of guess password attack explains num_failed_logins = 1 majority. Telnet sessions are used to access remote networked computers. Telnet sessions are not encrypted which gives adversary the benefit of sniffing the packet from the network.

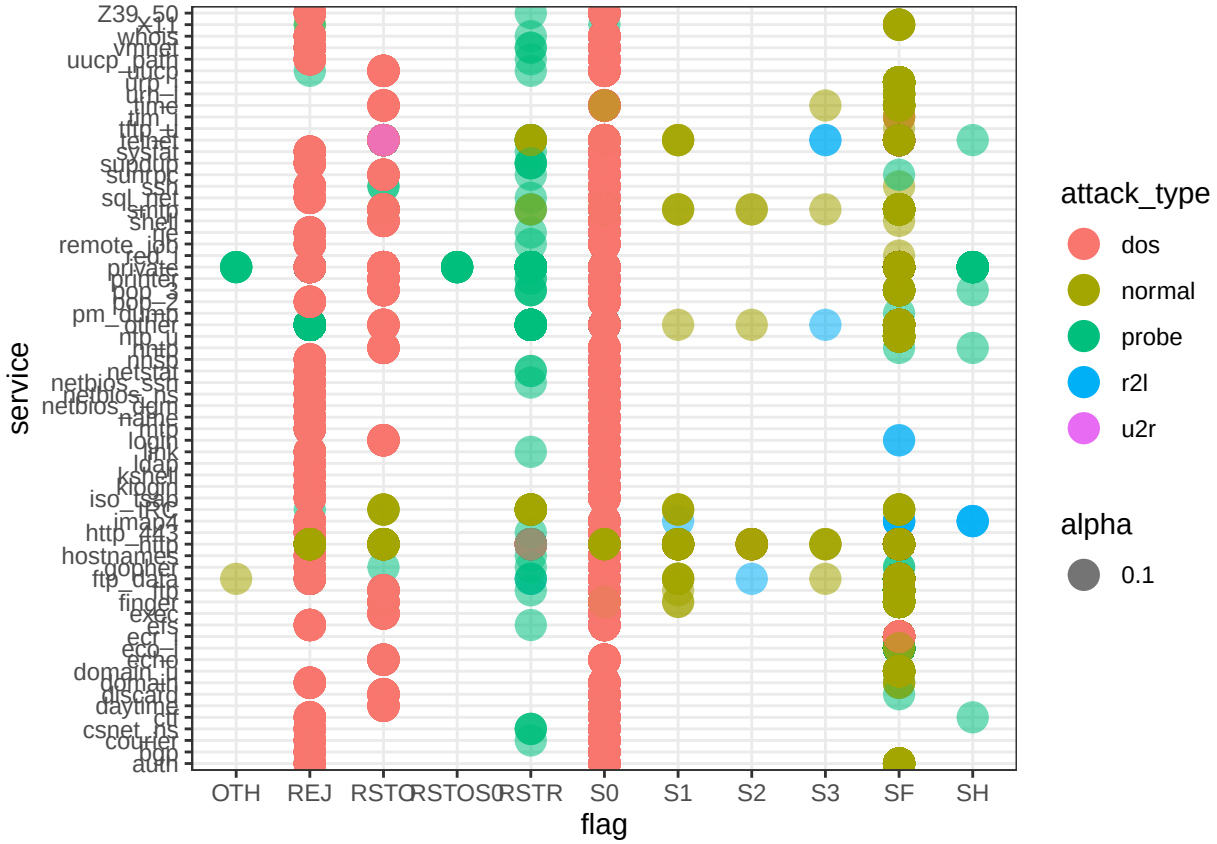
Flag Indicator

A flag is an ad-on information on the host side of the network. A flag is raised and recorded based on a every connection attempt. These are the following possible types of flags that can be raised:

- S0 – Connection attempt seen, no reply.
- REJ – Connection attempt rejected.
- S1 – Connection established but not terminated.
- SF – Normal establishment and termination.

- S2 – Connection established and close attempt seen by originator
- S3 - Connection established and close attempt seen by responder
- RSTO - Connection established, originator aborted
- RSTR - Connection established, responder aborted
- RSTOSO - Originator sent a SYN followed by a RST, SYN, ACK not seen by the responder
- OTH - No SYN seen, just midstream traffic
- SH - Originator sent a SYN followed by a FIN, SYN, ACK not seen by the responder. [16]

```
ggplot(data)+
  geom_point(aes(service,flag,color = attack_type,alpha = 0.1),size = 5)+coord_flip()+theme_bw()
```

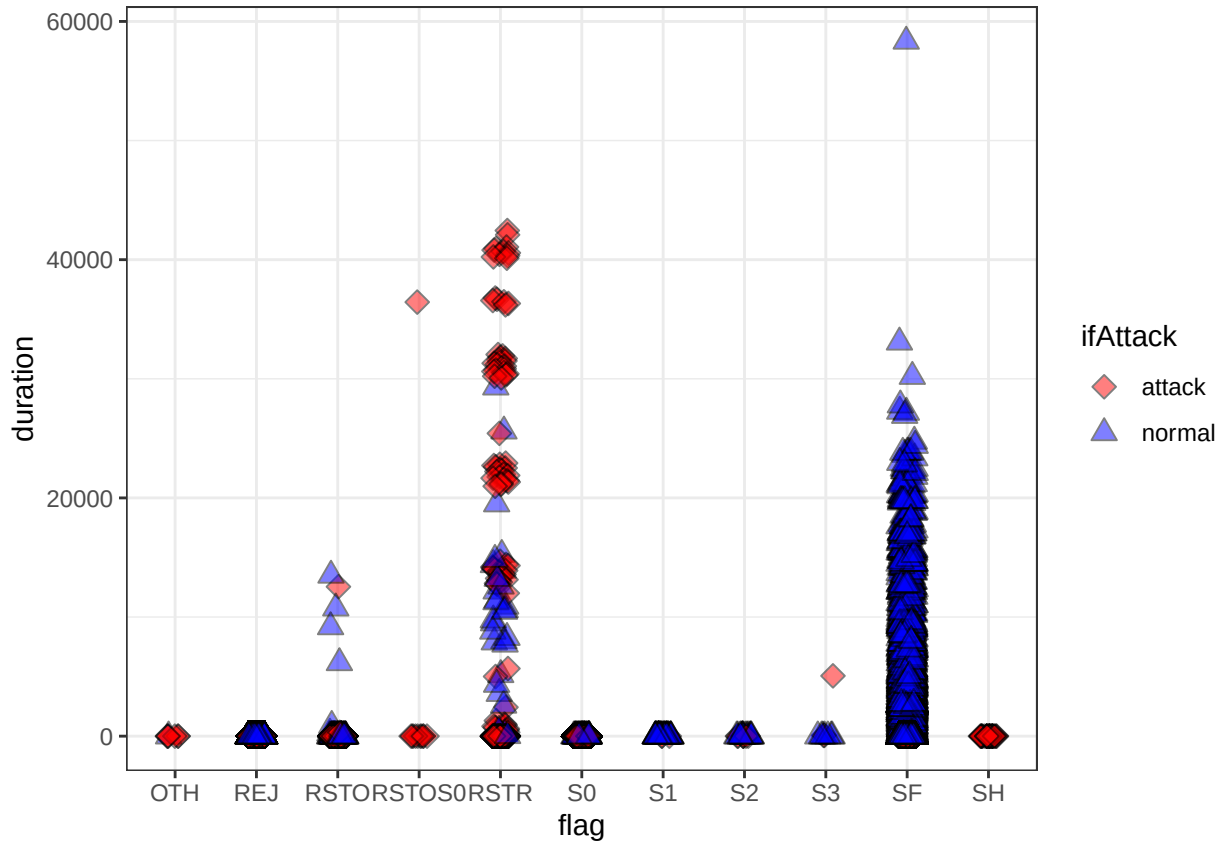


Observation:

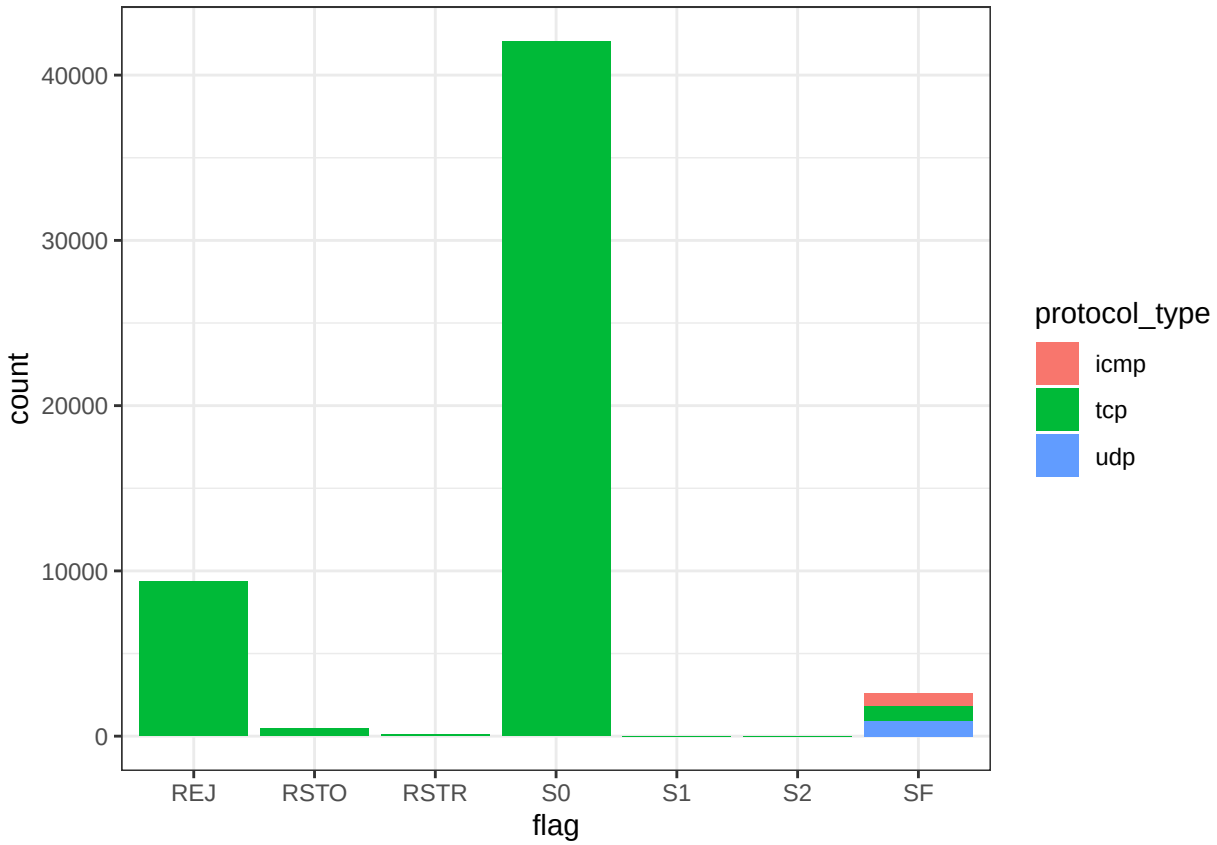
- S0 and REJ are strong indicator for DOS attack.
- RSTOS0 and RSTR are strong indicator for Probe attack.

As supported by recent research [cite], flag is a strong indicator that can identify malicious traits in the feature vector and categorize them into correct sub-category. From a top view, RSTOS0 and SH can distinguish between legitimate and malicious traffic.

```
ggplot(data, aes(flag,duration,  fill = ifAttack, shape = ifAttack)) +
     geom_point(position=position_jitter(h=0.1, w=0.1), alpha = 0.5, size = 3) +
     scale_fill_manual(values=c("red","blue")) +
     scale_shape_manual(values= c(23, 24)) +theme_bw()
```




```
filter(data, attack_type == "dos") %>% ggplot() +
  geom_bar(aes(x = flag, fill = protocol_type))+theme_bw()
```

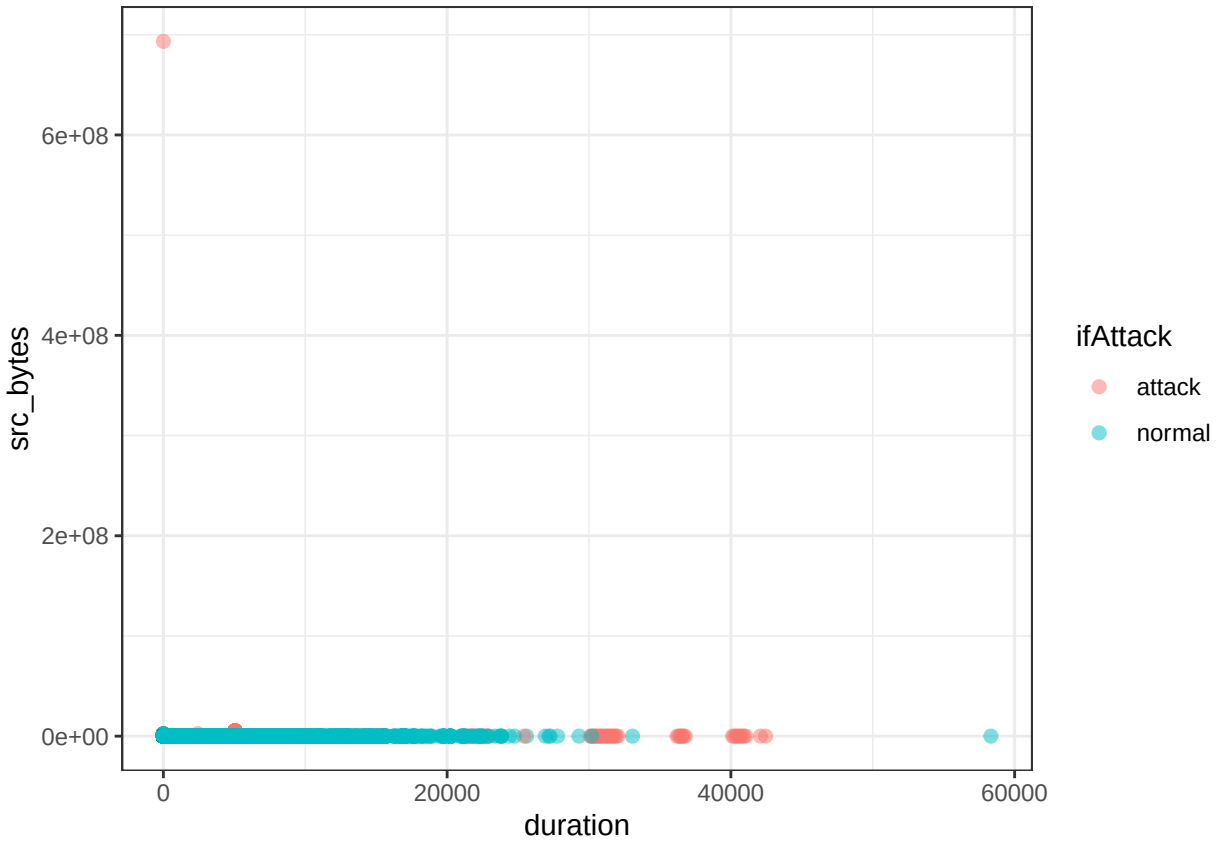


Observation: + To further support the hypothesis, S0, REJ, RSTO, RSTR, SF flags raised for DOS attack. + Interesting pattern seen for flag SF.

Duration Indicator

From my previous research experience, duration is an important indicator to identify malicious traffic. A traffic that has lived in the network for a long time on average can be deemed suspicious.

```
filter(data) %>% ggplot()+
  geom_point(aes(duration,src_bytes,color = ifAttack),size = 2,alpha = 0.5)+theme_bw()
```



Observation:

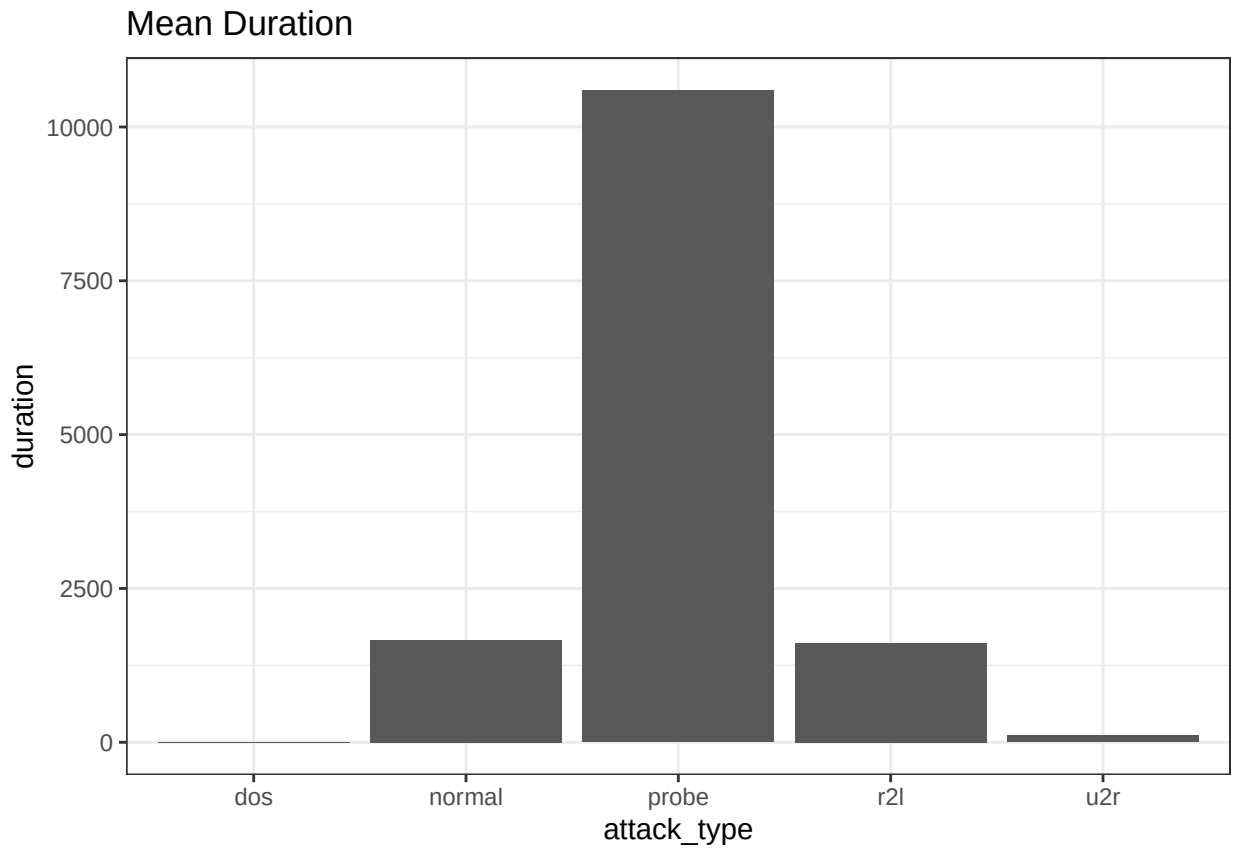
- Duration at more than 30,000 is malicious data.

Additionally, not all type of network attack take longer time to execute on average.

```
filter(data,duration !=0) %>% ggplot(aes(y = duration, x =attack_type)) +
  geom_bar(stat = "summary", fun.y = "mean")+labs(title = "Mean Duration")+theme_bw()
```

```
## Warning: Ignoring unknown parameters: fun.y
```

```
## No summary function supplied, defaulting to `mean_se()`
```



Observation:

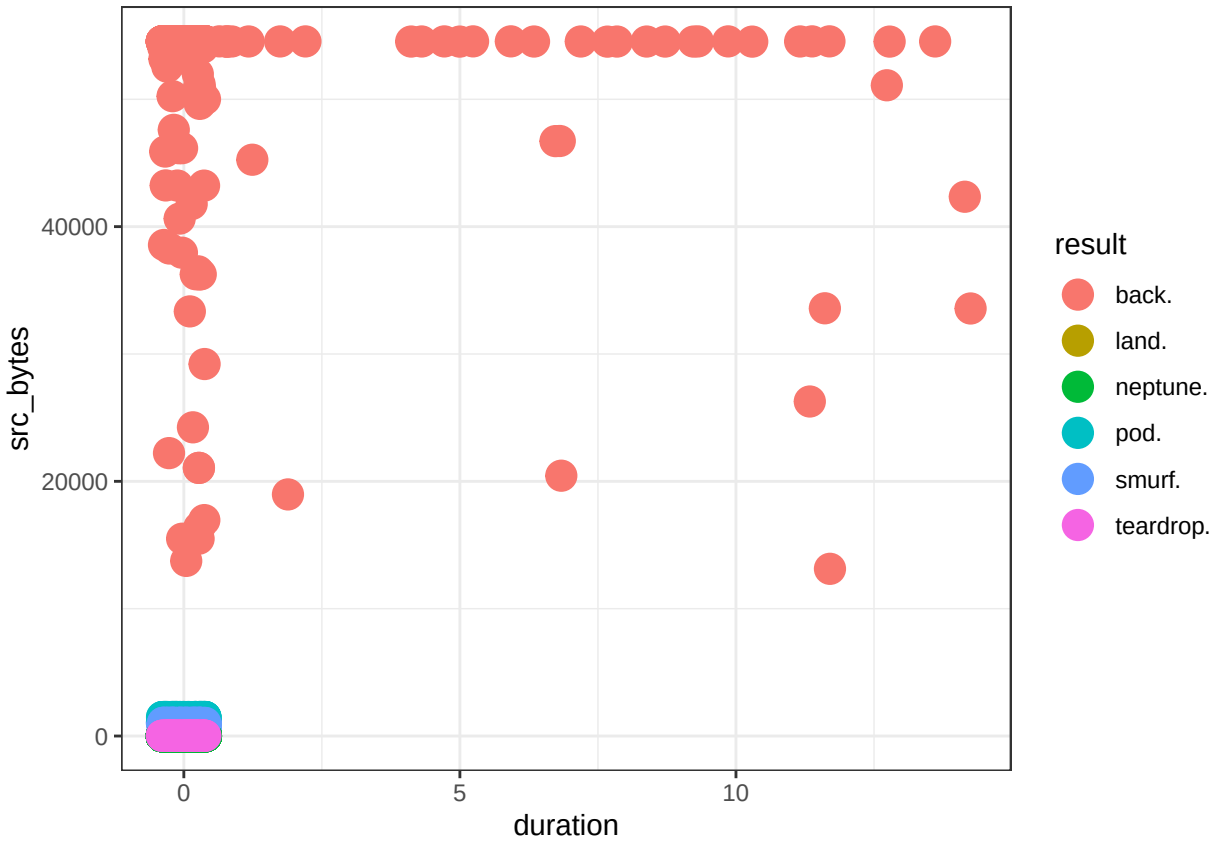
- DOS indicates a mean of duration 0. Possibly because the connection between host and the client becomes un available after the initiation of the DOS attack.
- Probe attack takes the longer duration on average.

DOS indicates mean 0 duration, hence such malicious packets can blend in with normal traffic.

Source Data Bytes Indicator

Source data bytes are request bytes sent by the client to host. In attempt to see the behavior of DOS records in the dataset, observe the relationship between source bytes (src_bytes) sent from client to host.

```
filter(data, attack_type == "dos") %>% ggplot()+
  geom_jitter(aes(duration,src_bytes,color=result),size = 5)+theme_bw()
```



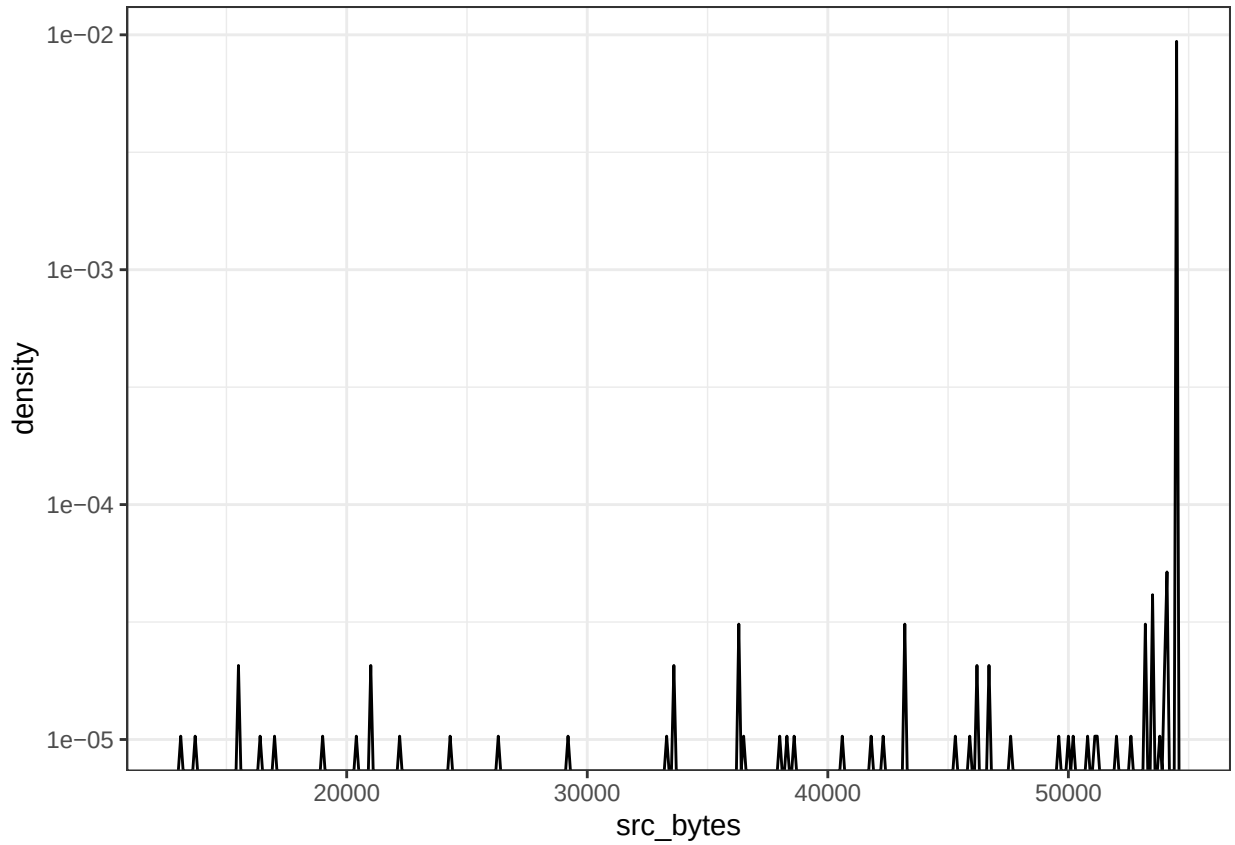
Observation:

- Unlike land. , neptune., pod., smurf. or teardrop., back. shows contrasting behavior.

Back. attack is a type of DOS which must be a buffer overflow attack. To prove this hypothesis, plot the density of source bytes.

```
filter(data,result == "back.") %>% ggplot(mapping = aes(x = src_bytes,y = ..density..)) +
  geom_freqpoly(mapping = aes(colour = srv_count), binwidth =100)+scale_y_log10()+theme_bw()
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

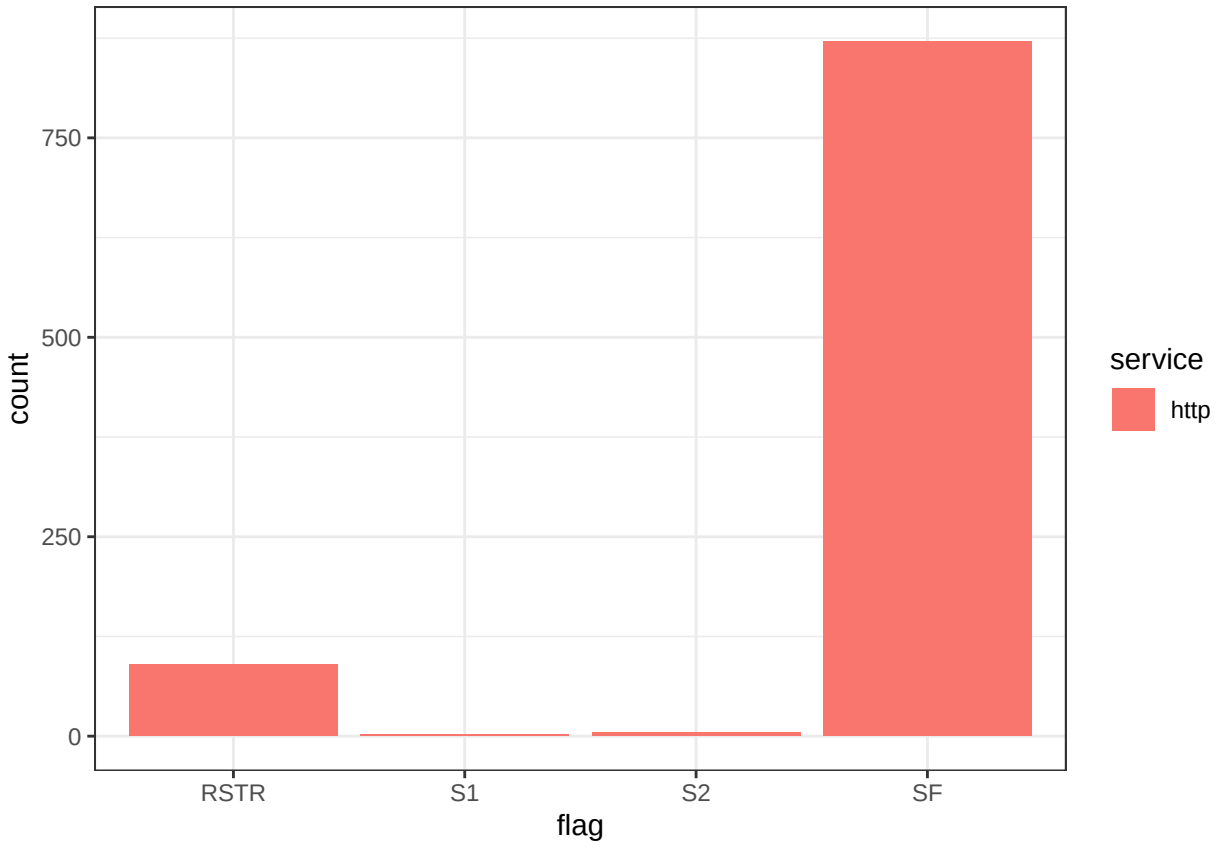


Observation:

- High density after 50,000 src_bytes.

To know more about the features of back. DOS attack:

```
filter(data,result == "back.") %>% ggplot() + geom_bar(aes(x = flag, fill = service))+theme_bw()
```



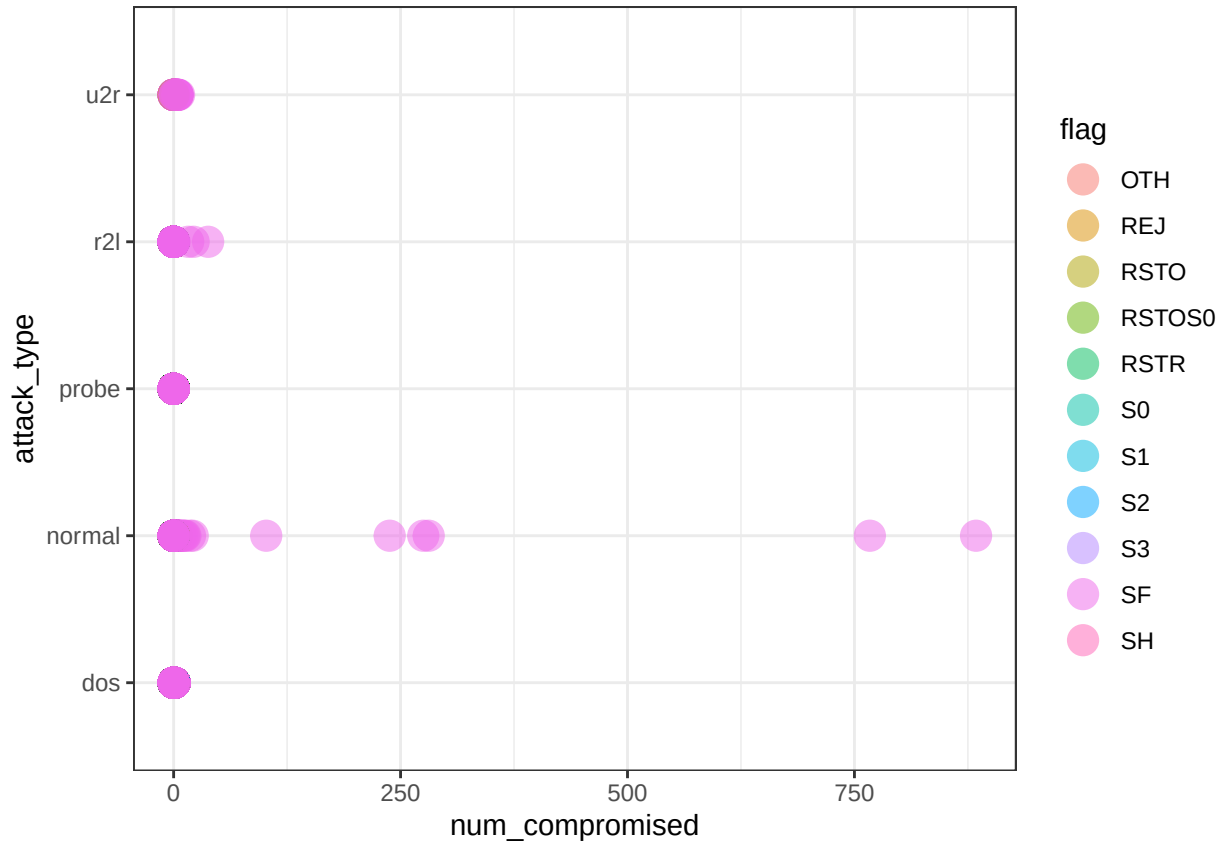
Observation:

- Back. DOS attack raises RSTR, S1, S2 flags.
- Exploited http service.

Number of Compromised Indicator

Number of compromised connection from the host side. The host is aware of malicious outbreak in the system and the host has terminated the connection.

```
ggplot(data = data, mapping = aes(x = num_compromised, y = attack_type, color = flag)) +
  geom_point( alpha = 0.5,size = 5)+theme_bw()
```



Observation:

- Normal traffic has highest num_compromised value.
- The flag raised is SF - Normal establishment and termination.

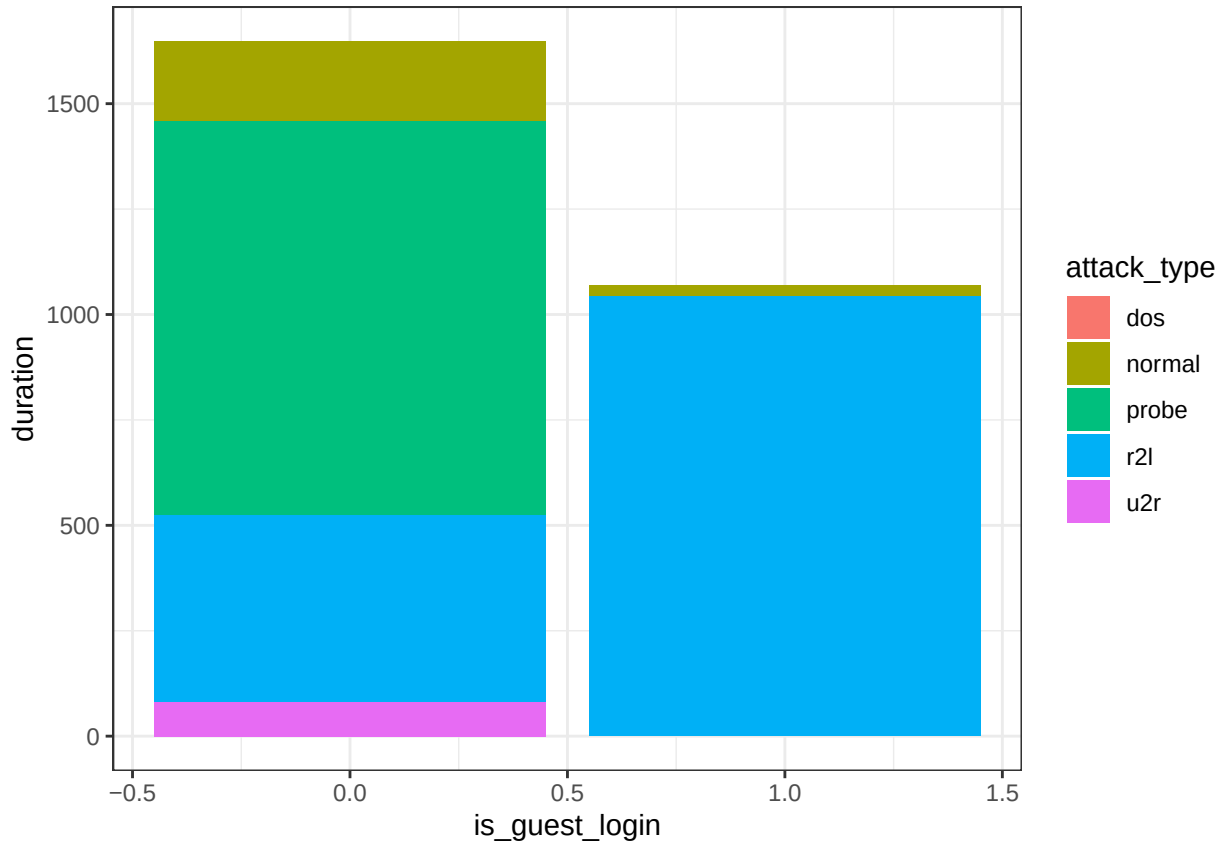
Guest login indicator

Guest login is a boolean data where 0 - yes, guest login, 1 - no, not a guest login.

```
ggplot(data=data, aes(x=is_guest_login, y=duration, fill = attack_type)) +
  geom_bar(stat = "summary", fun.y = "median")+
  theme(axis.text.x=element_text(angle=90,hjust=1))+theme_bw()
```

```
## Warning: Ignoring unknown parameters: fun.y
```

```
## No summary function supplied, defaulting to `mean_se()`
```



Observation:

- R2L - unauthorized access from a remote machine is the majority for is_guest_login = 1.

Correlation between Variables

Analyzing symbolic variables can be complex problem, in case there are so many feature vectors like KDD CUP 1999. A quick way to find a correlationship between the the feature vectors is using correlation matrix [2].

Correlational Matrix:

```
V1<- data[,1:1]
kdd_data2<- data[,5:41] #2-4 string data, not comparable
kdd_data <- cbind(V1,kdd_data2)
```

```
library(corrplot)
```

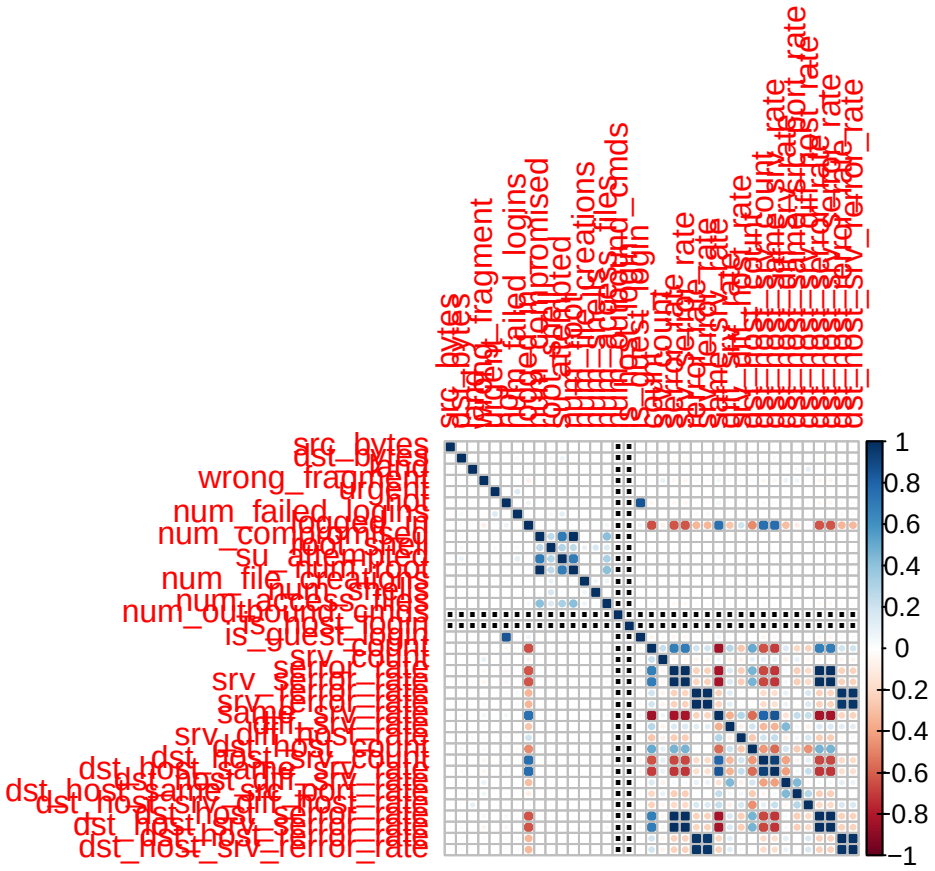
```
## Warning: package 'corrplot' was built under R version 4.0.5
```

```
## corrplot 0.84 loaded
```

```
correlation <- cor(data[,5:41])
```

```
## Warning in cor(data[, 5:41]): the standard deviation is zero
```

```
corrplot(correlation, method="circle", na.label= '.')
```



Observation:

- The dark blue points are directly related to each other.
- The red points indicate they are indirectly related to each other.
- The size of the points define how significant are the data.

Modeling

Random Forest Decision Tree is a widely used modeling technique for network intrusion detection systems. Random Forest Decision Tree does has some feature selection limitations as pointed out by previous researchers [1]. Using the indicator variables I analyzed, I will be modeling the data on my selected indicator/feature vectors.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.5
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
```

```
##     margin
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.5
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
mod_data<-data[,c("flag",
             "dst_bytes","src_bytes","num_compromised" ,
              "dst_host_srv_count","duration" ,"dst_host_same_src_port_rate","dst_host_diff_srv_rate" ,
              "dst_host_count","dst_host_srv_serror_rate","count",
              "dst_host_same_srv_rate","dst_host_srv_diff_host_rate","srv_count","srv_diff_host_rate","
```

```
control <- rfeControl(functions=rfFuncs, method="cv", number=10)
inTrain <- createDataPartition(y=mod_data$attack_type,p=0.1, list=FALSE)
training <- mod_data[inTrain,]
testing <- mod_data[-inTrain,]
dim(training)
```

```
## [1] 14562    19
```

```
model <- train(attack_type ~ .,method="rf",data=training)
model
```

```
## Random Forest
##
## 14562 samples
##    18 predictor
##     5 classes: 'dos', 'normal', 'probe', 'r2l', 'u2r'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14562, 14562, 14562, 14562, 14562, 14562, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9798367  0.9585618
##   15    0.9970296  0.9939976
##   28    0.9954086  0.9907263
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 15.
```

From previous results [1], random forest reaches the highest accuracy of 97.1% .

Using the specific features I analyzed, the highest achievable accuracy is 99.7%.

Conclusion

Network intrusion detection systems are built on blackbox machine learning algorithms which picks features that are deemed important. It is complex to analyze which feature is the important over the other so that the model is not overfitted or underfitted. Modeling this problem using random forest is a well traversed approach, but random forest essentially suffers from feature selection problem [1]. According to my high level analysis on network data, I have the following understanding of the selected features:

(1) Duration is an important indicator. A connection that lives in the network for a longer time is indeed suspicious. Although, training a machine learning model on only duration feature can result into some false positive malicious packets, i.e., it is evident from the plots I formed that malicious traffic don't always have to be in the network for a long duration.

(2) Failed attempt indicator helped in identifying which flavor of "guess password" attack the adversaries were using. I suspected it to be bruteforce attack, but in reality, adversaries exploited TELNET and sniffed the password from the network.

(3) Flag indicator is an obvious and important indicator.

(4) From guest login indicator, not all adversaries use guest login to attack the network.

(5) Source byte is an important indicator. Abnormally large size packets for a long duration of time can bring down a host (unavailable).

Random Forest shows a high accuracy of 99.7% using the features I used. Although, it is worthwhile noting that KDD CUP 1999 is an old data and the type of attacks has evolved ever since. On the concluding note, network data is complex. Depending on what type of attack detector we want, machine learning features needs to be tweaked. Source data byte is important feature for DOS attack, but duration indicator is not.

Reference

[1] https://www.irjet.net/archives/V4/i8/IRJET-V4I8137.pdf [2] https://cran.r-project.org/web/packages/corrplot/vignettes/corrplot-intro.html [3] http://kdd.ics.uci.edu/databases/kddcup99/ [4] https://arxiv.org/pdf/1903.02460.pdf