Zero Trust: Malicious Network Traffic Detector

Amie Davis

Bellevue University

Spring 2021

https://amodavis.github.io/

Abstract

As the amount of data we share increases, cybersecurity becomes more critical to protect it. We have also seen the need for accessibility increase during the pandemic. Users are connecting from multiple locations on multiple devices. It is necessary to provide this flexibility, while still securing our networks. To enable more access, we must review and validate all access points continually. Machine learning is critical to achieving this goal.

Zero Trust: Malicious Network Traffic Detector

Zero Trust is a cybersecurity framework which embraces the mantra “never trust, always verify.” There are multiple avenues that can be explored with Zero Trust. Identity and Asset Management is required to manage and secure human and non-human identities. Threat Protection is needed to provide automated security, stopping attacks before they begin. Information Protection is essential to protect sensitive data, and Cloud Security is necessary to protect assets stored in the Cloud. For this project, I will focus on Threat Protection where incoming and outgoing traffic is reviewed to identify suspicious network activity.

The framework of Zero Trust security is "built on four main pillars: (1) verify the user, (2) validate their device, (3) limit access and privilege, and (4) learn and adapt” (Columbus, 2018). With Zero Trust, machine learning can be used not only to identify normally accepted behavior but to allow users to “apply for conditional access without impacting user experience” (Columbus, 2018). This is considered ideal since all anomalies are blocked, but users can be provided automatic temporary access to move forward. Zero Trust architecture focuses on protecting resources, as opposed to the network. These resources include hardware, applications, data, and users. The architecture assumes there is an intruder inside the network and ensures resources cannot be taken or misused (Rose, et al, 2020).

There was a time when networks were blocked at the perimeter only. Once credentials were authenticated, users were allowed access. This is no longer sufficient. It is essential to review data at various checkpoints to assess for malicious activity. The traditional approach of blocking suspected traffic has changed to allowing only expected traffic under the Zero Trust model.

One such checkpoint is at the boundary where network traffic first enters your network. There, you can review traffic to determine whether it should be suspect, such as Tor traffic. This process can be repeated at each checkpoint. “Tor (aka The Onion Router) is software that allows users to browse the web anonymously by encrypting and routing requests through multiple relay layers or nodes" ("Alert," 2020). Tor was created for a common cause that "internet users should have private access to an uncensored web." Since its origins in the 1990s, Tor has been used to "route traffic through multiple servers and encrypt it each step of the way" ("The Tor Project," n.d.) However, a Joint Cybersecurity Advisory from the Cybersecurity and Infrastructure Security Agency (CISA) advises to monitor against Tor traffic, "While Tor can be used to promote democracy and free, anonymous use of the internet, it also provides an avenue for malicious actors to conceal their activity because identity and point of origin cannot be determined for a Tor software user" ("Alert," 2020).

Method

Business Problem

Increased system accessibility invites an increase in malicious cybersecurity threats. Allowing access from outside the network makes it more difficult to identify what network traffic should be allowed and what should be blocked. We need to allow personnel to work from remote locations and continue to be productive. Access is required whether using company devices over a Virtual Private Network (VPN) or a personal device. This has become the new normal during the pandemic, but we must continue to adhere to strict protocols and keep our systems secure.

Business Objective: *Identify network traffic that should be blocked, regardless of device or physical location.*

Data Sources

The data for this project come from the DarkNet 2020 dataset from the Canadian Institute for Cybersecurity at the University of New Brunswick. This dataset expands on Wireshark data previously collected in 2016, during which the data were analyzed and labeled as either malicious (Tor) or benign (non-Tor). What makes this dataset unique is that it also includes the category of the traffic: Audio-Stream, Browsing, Chat, Email, Peer-to-Peer, File Transfer, and Video-Stream. This provides a more readable dataset, as opposed to deciphering usage methods. In addition to source and destination ip addresses, ports and protocols are included. There are also approximately 75 numeric features describing the traffic, such as packet size and speed.

Exploratory Data Analysis

Python was used to prepare data for the initial statistical analysis and visualizations. This allowed me to get to know the data and the available features. I reviewed the dataset to determine data types and volume of data. Distributions were analyzed and correlation between features determined. Both Python and PowerBI visualizations were used to identify outliers and anomalies and address specific research questions.

I was surprised to find infinity values for Flow Bytes and Flow Packets. Per Oracle's IP Administration Guide, "A flow is a sequence of packets that are sent from a particular source to a particular, unicast or multicast, destination" ("Flow Labels," 2010).   If the size and number of packets in a flow are immeasurable by the router, they can be labeled as infinite. For these entries, I imputed a value 100 times the maximum value in the column. This allowed me to retain these flow records since they were valid. Null values were imputed with zero.

Features were reviewed for dependence and multicollinearity, removing fields as needed to maintain independence and minimize correlation. I assessed correlation to perform feature reduction. First, it was necessary to convert data into numerical features. I was able to do this by splitting ip address segments into separate fields which I then converted to integers. Highly correlated features, those with a Pearson’s coefficient greater than 0.8, were removed, reducing the number of features from 51 to 35. Principle Component Analysis (PCA) was also performed but was not used to for feature reduction due to a nonlinear separation.

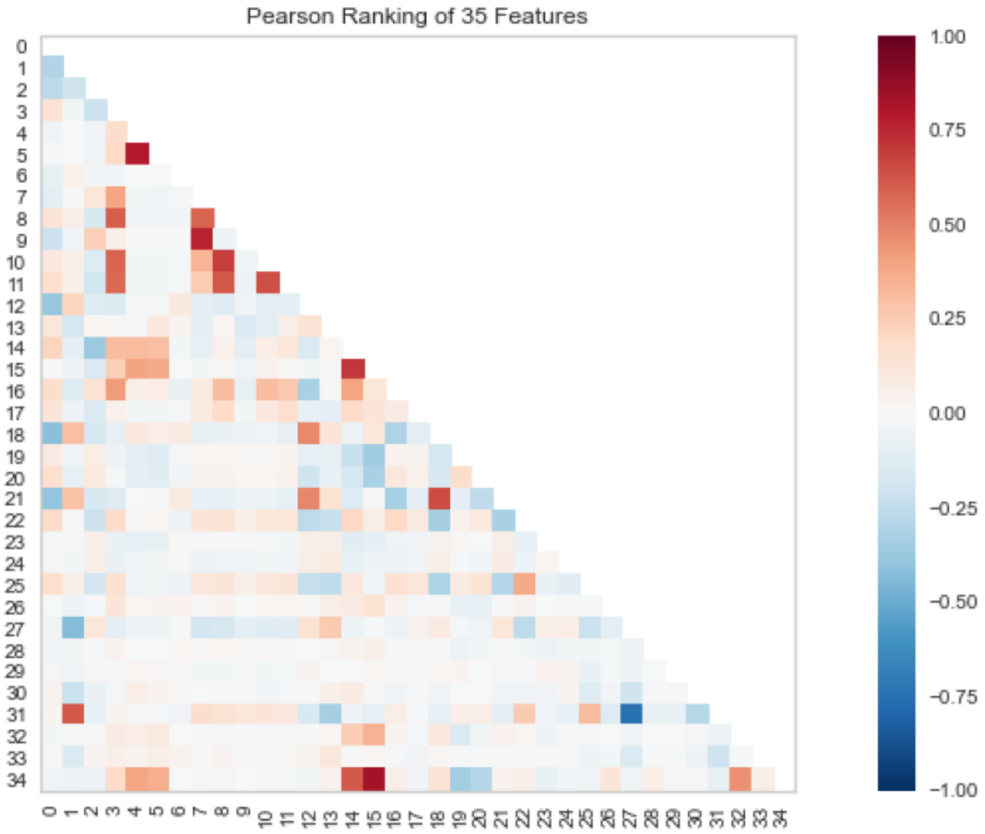


Figure 1: Correlation Matrix

Modeling Objective

So, how do we know if network traffic is considered Tor?

Modeling Objective:	*Perform supervised learning to predict whether network traffic is Tor.*

I used a logistic regression algorithm in Python to create a binary classification model to predict whether network traffic was Tor or not. Multiple models were built using different solvers and evaluated based on the number of false predictions, as well as the Matthews Correlation Coefficient (MCC). Accuracy is not the best measure for logistic regression with imbalanced classes. In this case, there is an imbalanced class of 2% Tor traffic versus 98% Non-Tor traffic which can elevate metrics, such as F1, precision, and recall. After reviewing multiple metrics, I believe that Matthews Correlation Coefficient is the best metric for this case. Weights were also added to help counter the imbalance in the model.

Results

Exploratory Data Analysis

The records of malicious traffic in the dataset are few in comparison to the benign traffic, less than 2%. Although this is a realistic representation, the imbalanced dataset required special methodology, as referenced in the Methods section above. This imbalance, however, is representative of what is expected in the population, so it is important the model be built to handle such an imbalance.

Most of the network traffic was in the Peer-to-Peer category, even more than Browsing traffic. Per Neagu (2019), peer-to-peer connections are most commonly used to download files from the internet. This includes Windows updates and online gaming. "Unfortunately, peer-to-peer networks are also commonly used for illicit activities," such as piracy.

Protocols for this network traffic were either Transmission Control Protocol (TCP), designated as protocol 6, or User Datagram Protocol (UDP), designated as protocol 17, for the most part. These are the two most common methods of communication over the internet (Doyle, 2018). Reviewing the source subnet data, a lot of the non-private traffic appears to have originated in from Canada or France. This aligns with the data source origin of New Brunswick, Canada.

Research Question Visualizations

PowerBI visualizations were used to review features and address questions formulated during my initial research.

- How do you know traffic is malicious?

You never know whether network traffic is malicious until a malicious action is taken. That is too late. Any traffic that is suspect should be blocked and let in by exception only after source and intent is verified. This project addresses only one type of malicious traffic, Tor. There are many alerts presented by CISA for a variety of concerns that can be detected and blocked but are outside of the scope of this project.

- What features contribute the most to Tor traffic?

I used scaled data and reviewed the model coefficients to approximate feature importance. Features with positive scores predict class 1 (Tor), whereas features with negative scores predict class 0 (Non-Tor). The feature that has the most impact on the model is Feature 1, the Destination Port, followed by Feature 32, the Voice Over IP (VOIP) indicator.

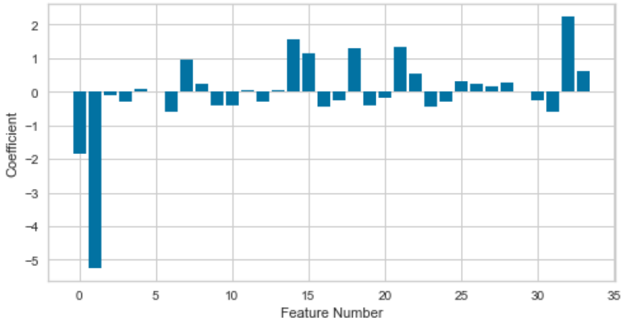


Figure 2: Feature Importance

- From what ip subnet is Tor traffic most likely to come?

Actual origin of Tor traffic cannot be determined. However, using an ip address locator ("IP Address," n.d.), I was able to identify the last identified jump point of the traffic labeled as Tor. Most of the Tor traffic in this dataset is marked as coming from a private Local Area Network (10.0.2.15). Those that did not, however, are identified as originating in Paris, France (195.154.107.23), New Brunswick, Canada (131.202.240.150), or Johnson City, Tennessee (198.52.200.39).

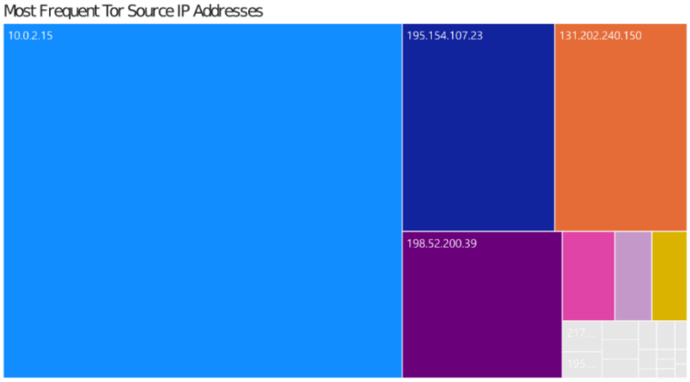


Figure 3: Tor Source Addresses

- How much Tor traffic is detected by each category?

Most of the Tor traffic was in the Voice Over IP or "VOIP" category which delivers voice over the internet, followed by Browsing, and then almost equal portions of Audio-Streaming, Peer-to-Peer connections, and Video-Streaming.

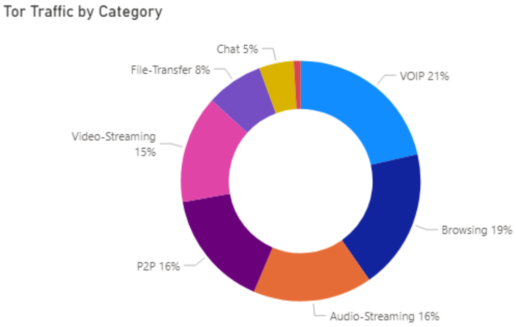


Figure 4: Tor Traffic Categories

- Is Tor traffic faster or slower than non-Tor traffic?

To deal with the questions on traffic speed, I selected to use Flow Duration. Flow Duration represents the length of the connection in seconds. These waterfall charts reflect the most frequent flow durations for non-Tor traffic and Tor traffic. The most common duration of non-Tor traffic is 16 seconds, whereas the most common duration for Tor traffic was over 7 minutes. Some common Flow Durations for Tor traffic were over 24 hours.

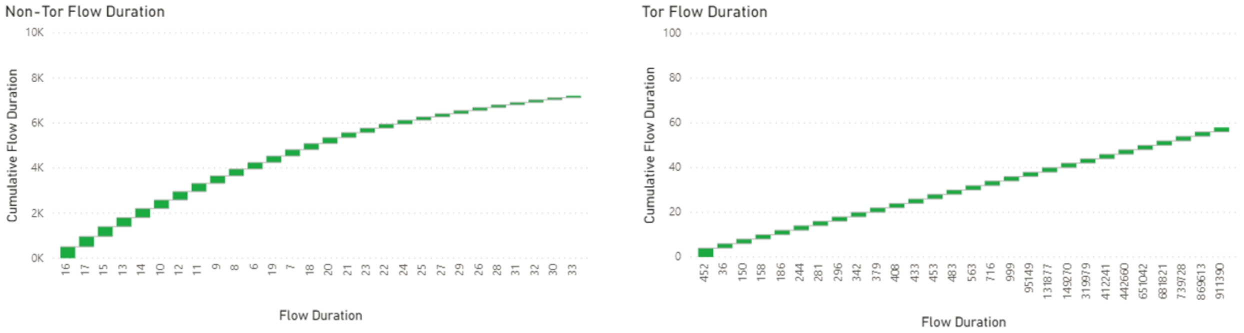


Figure 5: Flow Duration

- Is Tor traffic larger or smaller than non-Tor traffic?

To address questions on traffic size, I selected to use Flow Packets. Flow Packets are the number of packets in each data flow. Most packets ranged in size from 0.2 to 125,000 for non-Tor traffic and from 0.2 to around 60,000 for Tor traffic.

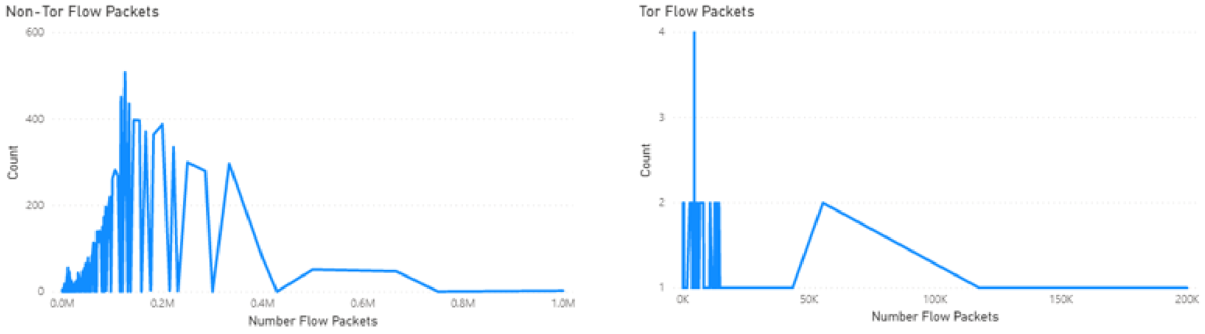


Figure 6: Flow Packets

Modeling

Scikit-Learn’s LogisticRegression algorithm was used to build the logistic regression model to predict if traffic was Tor or not. The class_weight parameter was set to ‘balanced’ to automatically adjust weights. Per the Scikit-Learn documentation (“sklearn.linear_model ,” n.d.), the weights are adjusted automatically to be “inversely proportional to class frequencies in the input data." Although standardization is not needed for Logistic Regression, it is recommended when using weights to counter imbalanced classes.

Multiple models were built with different solvers and evaluated to select the best solver for this use case. Accuracy is not the best measure for evaluation here since the target class is imbalanced. However, I used a confusion matrix to note the number of false predictions for each solver. See Table 1 below. To avoid over-fitting, sag or saga solvers are preferred, since the other solvers predicted no false negatives of the larger class.

Solver	Number False Negatives
lgfgs	0
liblinear	0
newton-cg	0
sag	1
saga	1

Table 1: LinearRegression Solver False Negatives

Matthews Correlation Coefficient (MCC) was used as the primary metric for solver comparison. Although the best MCC value resulted from using the newton-cg solver at 95%, to avoid overfitting, I selected sag as the solver for the final model with an MCC of 91%.

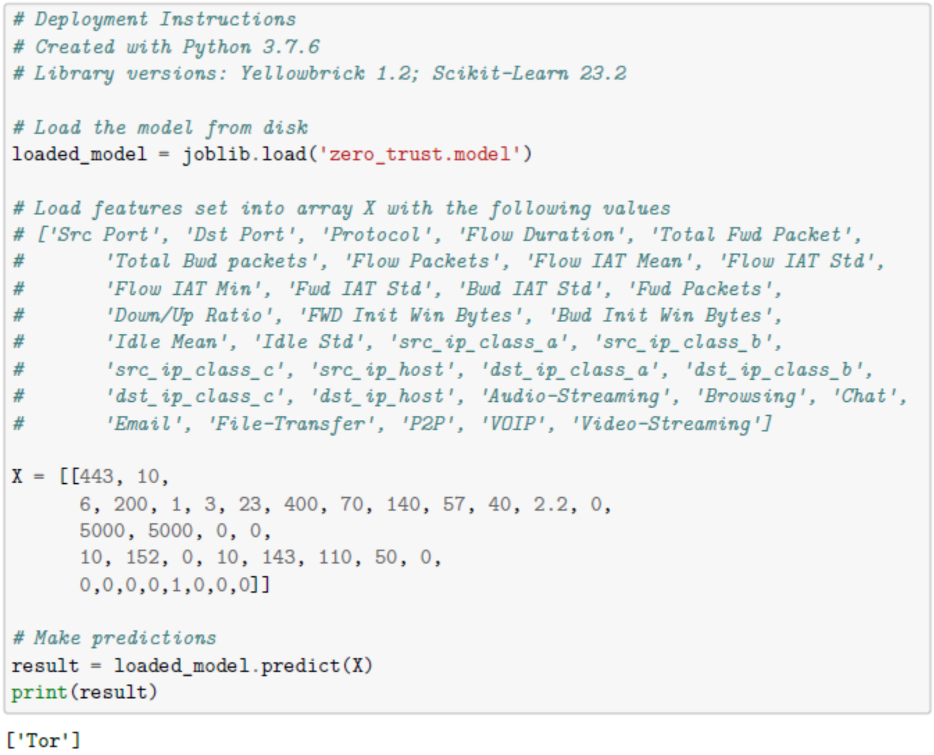
Solver	MCC	Number False Negatives
lbfgs	91%	0
liblinear	94%	0
newton-cg	95%	0
sag	91%	1
saga	89%	1

Table 2: LinearRegression MCC Comparison

After reviewing multiple metrics, I believe that Matthews Correlation Coefficient is the best metric for this case. Accuracy is not recommended as a logistic regression measure. The imbalanced class of 2% Tor traffic versus 98% Non-Tor traffic can elevate other metrics, such as F1, precision, and recall. With added weights to counter the imbalance in the model and the high scores in all metrics, including 91% MCC, I believe this is a good model and can be deployed.

Deployment

For deployment, load the saved model into your environment. You can then pass Wireshark data via an array to make predictions. The features passed should be limited and presented in the format as specified below. You can pass one record at a time or multiple records in a matrix. The model will return the label “Tor” for suspected Tor traffic and “Non-Tor” if the record is not suspicious. All Tor traffic should be blocked as soon as it is detected. See specific deployment instructions and sample below.



```
# Deployment Instructions
# Created with Python 3.7.6
# Library versions: Yellowbrick 1.2; Scikit-Learn 23.2

# Load the model from disk
loaded_model = joblib.load('zero_trust.model')

# Load features set into array X with the following values
# ['Src Port', 'Dst Port', 'Protocol', 'Flow Duration', 'Total Fwd Packet',
#       'Total Bwd packets', 'Flow Packets', 'Flow IAT Mean', 'Flow IAT Std',
#       'Flow IAT Min', 'Fwd IAT Std', 'Bwd IAT Std', 'Fwd Packets',
#       'Down/Up Ratio', 'FWD Init Win Bytes', 'Bwd Init Win Bytes',
#       'Idle Mean', 'Idle Std', 'src_ip_class_a', 'src_ip_class_b',
#       'src_ip_class_c', 'src_ip_host', 'dst_ip_class_a', 'dst_ip_class_b',
#       'dst_ip_class_c', 'dst_ip_host', 'Audio-Streaming', 'Browsing', 'Chat',
#       'Email', 'File-Transfer', 'P2P', 'VOIP', 'Video-Streaming']

X = [[443, 10,
      6, 200, 1, 3, 23, 400, 70, 140, 57, 40, 2.2, 0,
      5000, 5000, 0, 0,
      10, 152, 0, 10, 143, 110, 50, 0,
      0,0,0,0,1,0,0,0]]

# Make predictions
result = loaded_model.predict(X)
print(result)
```

```
['Tor']
```

Discussion

Limitations

This project was limited by the dataset provided. Rather than live data, the data were generated to protect security. Production Tor traffic logs are not available to the general public. This generated dataset may present different qualities. Since the data were also generated in a Canadian academic setting, the patterns identified may differ from the patterns in your corporate security logs.

This project addresses only one type of malicious traffic, Tor. There are many alerts presented by CISA for a variety of concerns that can be detected and blocked but are outside of the scope of this project. Similar methodology can be applied to detect other vulnerabilities.

Future Recommendations

It is recommended to add this model into your production pipeline to stream data in real-time from network traffic logs. To facilitate swift reaction, streaming is recommended over batching. When Tor traffic is detected, traffic should immediately be blocked at the firewall, notice of rejection returned, and details logged for future investigation. This should be repeated at all checkpoints. If a user is blocked from accessing your network, they should be provided a method to request an override which could allow limited access only.

Conclusion

Zero Trust is a framework, not a solution. Data must be checked at every checkpoint, not just at the boundary. Assume that all traffic is suspect. It is no longer alright to only check at the boundary and assume you have caught everything. As the mantra goes, “never trust, always verify.” It will be necessary to place detectors at each firewall within your enclave. Don’t assume that traffic that has been classified as benign previously will continuously be benign. For this reason, models must continually be updated and retrained frequently. There is a delicate balance between accessibility and security, but security must prevail.

References

Anchan, P. (2019, December 6). *Achieve Smart Security Using Machine Learning for Zero Trust: Ilantus Blog*. Ilantus Technologies. https://www.ilantus.com/blog/machine-learning-for-zero-trust-how-can-it-be-done/.

Alert (AA20-183A): Defending Against Malicious Cyber Activity Originating from Tor. Cybersecurity and Infrastructure Security Agency CISA. (2020, October 24). https://us-cert.cisa.gov/ncas/alerts/aa20-183a.

Chicco, D., & Jurman, G. (2020, January 2). *The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation*. BMC Genomics. https://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-019-6413-7.

CIC-Darknet2020. University of New Brunswick. (n.d.). https://www.unb.ca/cic/datasets/darknet2020.html.

Columbus, L. (2018, May 11). *Three Ways Machine Learning Is Revolutionizing Zero Trust Security*. Forbes. https://www.forbes.com/sites/louiscolumbus/2018/05/11/three-ways-machine-learning-is-revolutionizing-zero-trust-security/?sh=4a4d654481ed.

Doyle, S. (2018, December 17). *TCP vs. UDP: Understanding the Difference*. Privacy News Online by Private Internet Access VPN. https://www.privateinternetaccess.com/blog/tcp-vs-udp-understanding-the-difference/.

Flow Labels (IPv6 Administration Guide). Oracle. (2010). https://docs.oracle.com/cd/E19683-01/817-0573/chapter1-26/index.html.

IP Address Geolocation Lookup Demo. IP2Location. (n.d.). https://www.ip2location.com/demo.

Neagu, C. (2019, November 26). *What are P2P (peer-to-peer) networks and what are they used for?* Digital Citizen. https://www.digitalcitizen.life/what-is-p2p-peer-to-peer/.

Redekop, D. (2017, July 28). *Using a Zero Trust Model to block outbound VPN, Proxy, TOR, and P2P*. DNSthingy. https://www.dnsthingy.com/2017/07/using-a-zero-trust-model-to-block-outbound-vpn-proxy-tor-and-p2p/.

Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020, August 10). *Zero Trust Architecture*. NIST. https://www.nist.gov/publications/zero-trust-architecture.

sklearn.linear_model.LogisticRegression. scikit-learn. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

The Tor Project: Privacy & Freedom Online. Tor Project. (n.d.). https://www.torproject.org/about/history/.

Appendix A

Presentation Q&A

Q: Why did you select Matthews Correlation Coefficient as your metric?

A: A colleague pointed me to an article in BMC Genomics (Chicco & Jurman, 2020) which highlighted the advantages to MCC over Accuracy and F1 measures in binary classification problems. MCC has been proven to provide the best assessment of all four quadrants of the confusion matrix, to include true positives, false positives, true negatives, and false negatives.

Q: Why didn't you reduce your features after performing PCA?

A: When graphing Principal Components 1 and 2, I discovered a non-linear separation in the target class. kernelPCA is available for reducing linearly inseparable data. However, I received memory errors attempting to use that kernel with my limited processor, so I elected to retain all 35 features.

Q: Why did you choose to keep the records with infinity values rather than remove them?

A: It is common for routers to stop calculating packet settings at a certain point. This maximum setting is normally configurable. I chose to keep these records, since they are expected in other datasets. Since it is unknown what settings are used on the router, I opted to set the infinite values to 100 times the maximum value in that field.

Q: Did you identify any outliers?

A: There were outliers identified in Flow Packets and Flow Duration, but research revealed these to be realistic. Such outliers are expected in production data, so I opted to retain all records.

Q: Do you believe having an imbalanced dataset skewed your results?

A: No, by balancing the class weighs while building the model and using Matthews Correlation Coefficient as a metric, I believe I have mitigated the concern of the imbalance dataset.

Q: Why didn’t you select the solver with the highest MCC percentage?

A: Although the newton-cg solver had the highest MCC percentage at 95%, I was not comfortable using a setting that produced no false negative results with the test dataset with concern for over-fitting. Instead, I opted for the solver with highest MCC percentage with at least one false negative.

Q: Is there a reason you did not try Scikit-Learn’s lbfgs solver for the LogisticRegression method?

A: I was unable to use the default lbfgs solver along with the class weight setting, so that solver was not considered for final consideration. I did add to the comparison tables to eliminate confusion. Due to a known bug with the lbfgs solver code (corrected in Scikit-Learn 0.24 which is not yet compatible with Yellowbrick), I built models with multiple solvers to select best solver for this use case, while using the class weight setting.

Q: Did you consider using an artificial neural network model?

A: No, since the logistic regression model performed well, I did not feel the need to utilize more complex models.

Q: After assessing feature importance, why did you continue to use all features in your model?

A: Since most of the coefficients were low and relatively equal, with the exception of the two important features mentioned, I did not reduce features further.

Q: Did you experience any performance issues while building your model?

A: No, the models ran fairly quickly, regardless of the hyperparameters chosen. This enabled me to re-build the model multiple times with various hyperparameters settings for comparison.