Network Malware Detection

Jacob Hawkins
University of Tennessee
Dept. of Electrical Eng
and Computer Science
jhawki41@vols.utk.edu

Dylan Lewis
University of Tennessee
Dept. of Electrical Eng
and Computer Science
dlewis37@vols.utk.edu

Riley Crockett
University of Tennessee
Dept. of Electrical Eng
and Computer Science
rcrocke6@vols.utk.edu

***Abstract*—In this project, we analyzed network traffic data that contained information about a number of network connections between Internet of Things devices. We applied machine learning to this data with the goal of determining which of these connections represented a malicious attack and which were normal network connections. The goal of this was to investigate whether or not cyberattacks can be detected using only easy-to-gather network traffic data as opposed to more advanced methods. In addition to this, we wanted to see what features of a connection are most important when determining if a connection is malicious or benign. Based on our findings, malicious network traffic can be detected reliably by using machine learning methods. A variety of different methods produced accurate results, and only a few methods performed poorly.**

***Index Terms*—Network security, malware, machine learning, random forest model.**

I. Introduction and Motivation

Current estimates say that 7700 petabytes of data travel across the Internet everyday. This is an unfathomably large amount of data, and this amount is only going to continue to increase. One reason that there is so much network traffic these days is the rise of Internet of Things (IoT) devices. These are devices such as smart fridges, smart lamps, Amazon Alexas, and most 'smart' appliances. Devices such as these send and receive data over the Internet just like a phone or laptop, and they are doing it constantly.

There have been several major cyber attacks involving IoT devices. One example of this is the Verkada breach in 2021. Verkada is a cloud-based video surveillance company and the feeds of over 150,000 cameras were leaked. Also, in 2010, it was discovered that IoT devices in uranium enrichment facilities in Iran had been infected by a virus called Stuxnet created by the NSA [2].

These IoT devices have made the need for secure software more dire than ever as these devices can be essential to the households that use them, and they can deal with very private data. Since these devices are connected to the Internet, they are vulnerable to cyber attacks like any other computer. Securing IoT devices is one of the biggest areas of cybersecurity today, and that's why we chose to focus on this topic for our project.

II. Data Set

The data set used for this project came from Kaggle, and the data was collected by Stratosphere Labs [1]. This data set had several issues that will be discussed, but after removing all of the rows with null values there were 16,679 data points used in our analysis. Each of these data points contains information about a network connection including the duration, protocol used, ports, bytes sent and received, and several others. All of this data is very easy to collect and does not require any special or expensive tools.

The data collected from three IoT devices. These devices are a Phillip's HUE smart LED lamp, an Amazon Echo home intelligent personal assistant, and a Somfry smart doorlock. While all three of these are IoT devices, they are all products of different companies and they all serve very different household purposes [1].

Each data point was labeled as a malicious or begin connection. Originally, data points that were labelled as malicious in the data set from Kaggle were labelled had some extra detail in the label. For example, a connection would be labelled as "Malicious DDoS" or "Malicious MitM." We decided to change these labels to be only "Malicious" because we wanted to treat this as a binary classification problem.

A. Data Set Makeup

After the re-labeling, the data still had a makeup 64.58% malicious labels and 35.42% benign labels. Since this was a roughly one-to-three ratio, we wanted to balance our data. The makeup of the data before balancing is shown in Figure 1.

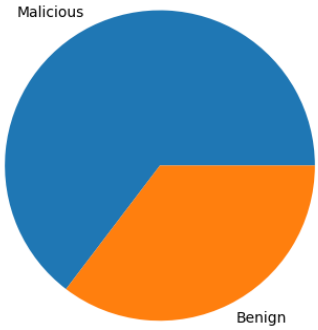

Fig. 1. Label distribution of prebalanced data

To balance the data, we used Pandas' DataFrame.sample() function to randomly select a number of malicious data points equal to the number of benign data points. The makeup of the data set after the balancing transformation is shown in Figure 2.

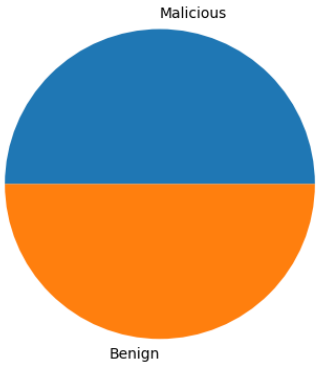

Fig. 2. Label distribution of balanced data

Certain features were dropped from our data set. The 'tunnel' feature indicated whether a connection was part of a network tunnel and was null for nearly all connections. The 'orig_local' and 'resp_local' features indicated if a connection originated from or received a response on a local network, it was also null for most data points. The 'id.orig_h' and 'id.resp_h' features contained the IP address of the origin and destination of the connection. These features were dropped because malicious traffic could originate from any IP and be sent to any IP and using them would likely make our models generalize poorly to other data sets.

B. Feature Correlation

After the data set was cleaned, we created a correlation matrix using our data set to see which features, if any, were dependent on one another. The correlation matrix, shown in the figure below, highlighted some features that were extremely correlated.

As the correlation matrix shows, there are very strong correlations between multiple pairs of features. The features pairs 'orig_pkts' and 'orig_resp_bytes,' and 'resp_bytes' and 'resp_ip_bytes' with both pairs having a correlation coefficient of 0.97. These correlations make sense, the number of packets sent from the origin would make the number of bytes sent from the origin increase. Similarly, it makes sense that the number of bytes sent by the receiver is correlated with the number of bytes the origin receives from the receiver. There are a couple other pairs of features that are correlated for this same reason, the amount of what was sent is the same as what was received.

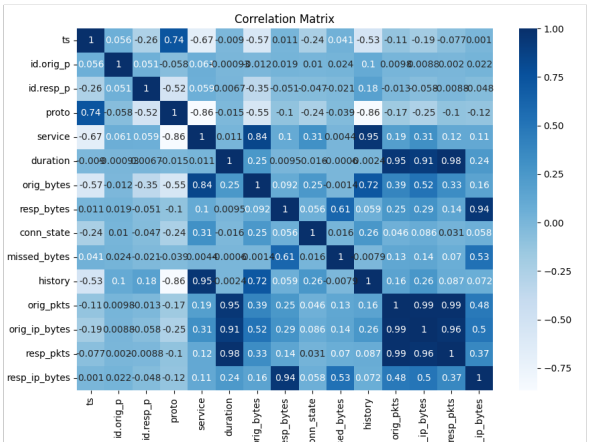

Fig. 3. Correlation matrix of features in data set

C. PCA

Given the complexity and high dimensionality of our data set, we turned to Principal Component Analysis (PCA). By analyzing the magnitude of contributions to the principal components, we could pinpoint which attributes were most influential in defining network behavior. This insight was invaluable in understanding the underlying patterns and relationships within our data.

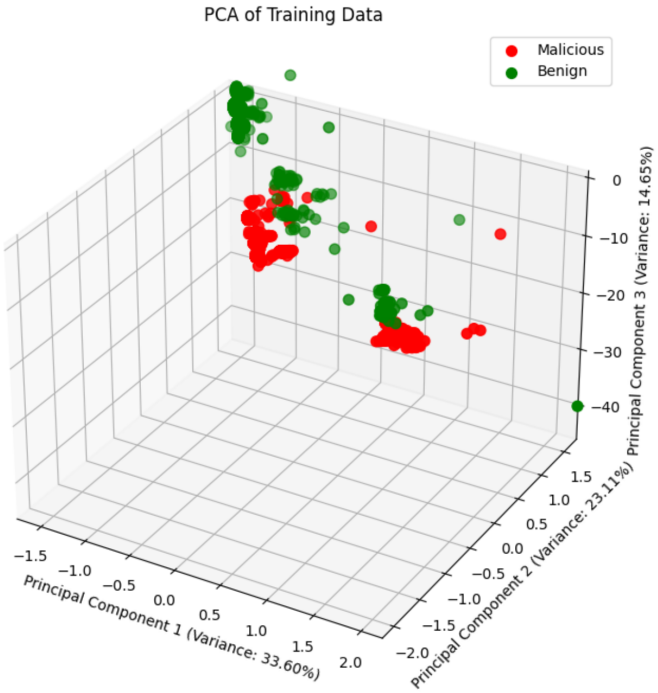

Fig. 4. PCA of Training Data

For the model generation, shown in Figure 4, features in the data set were standardized, then PCA was performed on the standardized features, reducing the dimensionality to three principal components. There was a small set of outliers in

the data, which was skewing the display of the model. Axes limits were adjusted based on percentiles to exclude outliers and better visualize the main cluster of data.

For PC1, the features orig_ip_bytes, orig_pkts, resp_pkts, and orig_bytes have the highest positive contributions, which implies that these features are the most influential in the direction of PC1. PC1 may be interpreted as a component that reflects overall traffic volume or activity level, as these features are often related to the amount of data transferred and the number of packets in network traffic.

For PC2, the features service, proto, and history have the highest magnitude contributions but with different signs, indicating that PC2 captures variance in the data associated with the type of service, the protocol used, and the connection history. The negative sign for service and history suggests that they vary in the opposite direction to proto.

PC3 is dominated by features related to bytes (resp_bytes, missed_bytes, resp_ip_bytes) with negative contributions. This might reflect aspects of the network traffic that relate to response sizes and missed data, potentially indicating retransmissions or incomplete transactions.

III. Machine Learning Approaches and Methodology

We began by splitting the data into an X matrix of all the data from the csv and y vector of the labels from the data. From there we used train_test_split from scikit-learn to split the data into a testing and training set with the random seed of 42 and a split test size of 0.33.

A. Approaches

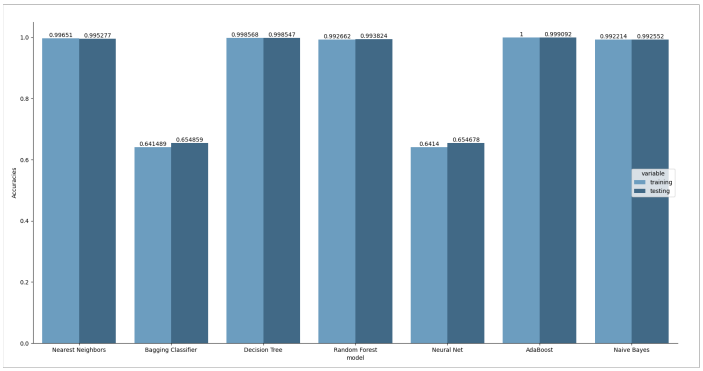

Fig. 5. Models testing and training accuracies

After splitting the data, we began testing on a variety of models. These included K-Nearest Neighbors, Decision Trees, Neural Networks, Bagging Radial Basis Function Support Vector Machine (RBF SVM), Naive Bayes, Ada Boost, and Random Forest models. These all performed relatively equal to one another averaging about 99% testing accuracy with randomly specified hyperparameters. The only ones that performed notably differently were the neural network at a testing accuracy of 49% and the bagging radial basis function support vector machine (RBF SVM) at a testing accuracy of 51%. Upon seeing this, we decided to proceed with the random forest model.

B. Random Forest Models

The random forest model consists of a specified number of decision trees that are fixed in structure and depth but contain random features. These features are typically chosen with replacement, meaning a single feature may be chosen multiple times in a tree. The leaves are then filled in with the training data. The testing data is filtered through each tree and a decision is made for that specific tree. The final decision is reached by allowing each tree to vote which classification it predicted. The classification with the most votes is chosen.

Benefits of using a random forest model include reducing the risk of overfitting the data [3]. Often just using a single decision tree can cause overfitting because of how well fitting they are to the training data. This can be mitigated by using multiple decision trees that lowers the variance and prediction error. Random forests also allow the user to easily see which features are the most important. It can be seen by comparing the difference of the model's accuracy when a specific feature is included or excluded.

Random forests can also present a challenge of computational efficiency [3]. As data sets grow and the number of decision trees required increases, computation time becomes a concern. This also means that more computational power is required when running them. Since the model creates multiple decision trees, each tree must be evaluated individually before summing the decisions and creating a final result. In addition to this, having multiple decision trees increases complexity when trying to interpret what the model is doing. A single decision tree is much easier to understand than a whole forest.

C. Hyperparameter Tuning

After selecting the random forest model, we began hyperparameter tuning. In doing so, we tested the number of estimators, or the number of trees created, and the maximum depth each tree can reach. For both of these parameters, the values 5, 10, 50, 100, and 200 were used.

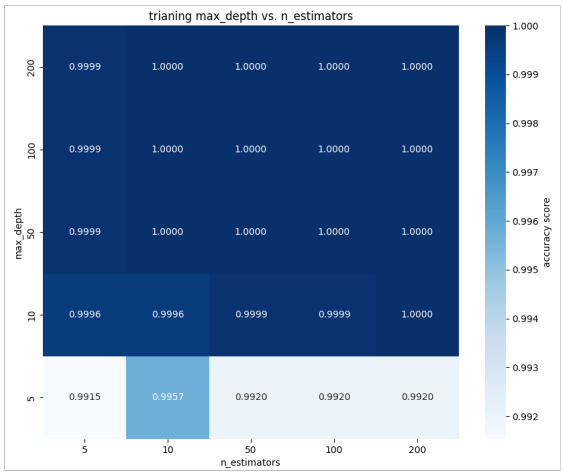

Fig. 6. Matrix of training accuracies during hyperparameter testing

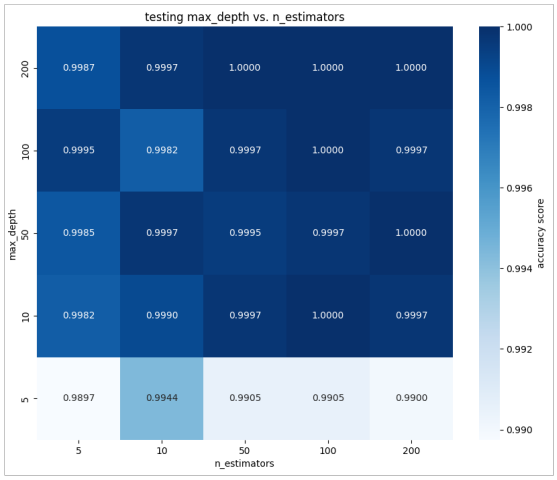

Fig. 7. Matrix of testing accuracies during hyperparameter testing

As shown in Figure 6, we began to see higher accuracy levels consistently at a maximum depth of ten and above. It is also interesting to note that the number of estimators does not significantly change the training accuracy. It has a very marginal effect going from five to 200. Similarly to the training accuracies, the testing accuracies, shown in Figure 7, began having a higher consistent score at a maximum depth of ten and above. Contrastly to the training data, the accuracies are less consistently 100%. Accuracy ranges, at a maximum depth of ten or above, from 0.9987 to 1.0. It appears that the number of estimators does impact the testing accuracy scores for the testing data.

After reviewing the testing and training accuracies with the variety of hyperparameters, we decided to choose a maximum depth of ten and the number of estimators to be 100. With these parameters, the model was able to predict with 100% accuracy on both the training and testing accuracy. We also chose this combination because it was the smallest number of both parameters that could achieve this accuracy score which, in result, creates a faster and more efficient model.

IV. Results

Based on our findings, machine learning models are good tools to use to inspect network traffic for malicious data. Our Random Forest model was able to make predictions about the testing data with 100% accuracy. The confusion matrix shown below in Figure 8 shows this performance.

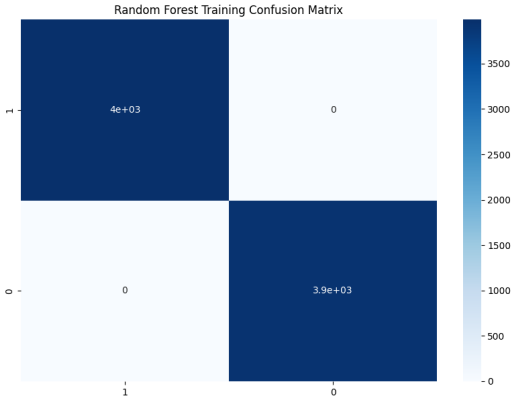

Fig. 8. Training confusion matrix

This random forest model performed extremely well on the chosen data set. Through hyperparameter tuning, we determined that using a maximum depth of ten captures the needed information and distinctions between features without overfitting the data. We also determined that 100 decision trees were needed to increase the accuracy from 99% to 100% and allowed the model to capture different edge cases that might be present in the data set. An example of one of the decision trees used by our Random Forest model is shown below in Figure 9. The maximum depth can be seen with in the figure.

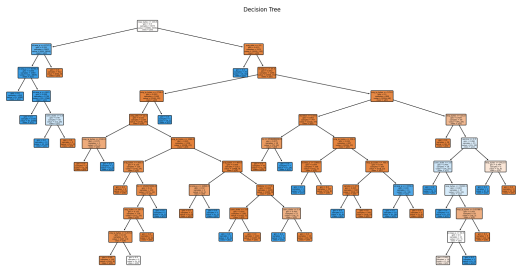

Fig. 9. Example of decision tree in Random Forest

A. PCA Results

Our group also wanted to determine what the most important feature are when determining if a network connection is malicious or benign. To do this, we analyzed the results of our principal component analysis

PC1	PC2	PC3
orig_ip_bytes: 0.3904	service: -0.3753	resp_bytes: -0.5864
orig_pkts: 0.3656	proto: 0.3752	missed_bytes: -0.5155
resp_pkts: 0.3408	history: -0.3734	resp_ip_bytes: -0.4800
orig_bytes: 0.3399	ts: 0.3276	duration: 0.2712
duration: 0.3057	duration: 0.3170	resp_pkts: 0.2058
service: 0.3009	resp_pkts: 0.2987	orig_pkts: 0.1423
proto: -0.2727	orig_pkts: 0.2820	orig_ip_bytes: 0.1184
history: 0.2714	orig_ip_bytes: 0.2402	orig_bytes: 0.0609
resp_ip_bytes: 0.2390	resp_ip_bytes: 0.2006	conn_state: -0.0406
ts: -0.2233	orig_bytes: -0.1913	proto: 0.0387
resp_bytes: 0.1542	conn_state: -0.1488	ts: -0.0330
conn_state: 0.1035	id.resp_p: -0.1265	service: -0.0308
missed_bytes: 0.0900	resp_bytes: 0.1262	id.orig_p: -0.0194
id.resp_p: 0.0228	missed_bytes: 0.1152	id.resp_p: 0.0152
id.orig_p: 0.0116	id.orig_p: -0.0215	history: -0.0145

TABLE I
FEATURE CONTRIBUTIONS

The identification of principal components and their associated features holds profound implications. PC1, dominated by features such as orig_ip_bytes, orig_pkts, resp_pkts, and orig_bytes, is indicative of the overall volume or activity level in network traffic. This component's focus on data transfer and packet numbers is crucial for understanding network load and identifying potential anomalies or inefficiencies in data flow.

PC2's composition, highlighting features like service, proto, and history, sheds light on the variance associated with the type of service, protocol used, and connection history. The contrasting contributions of these features reveal a nuanced understanding of how different types of network services and protocols interact and influence overall network behavior. The negative correlations for service and history with proto suggest a complex interplay that could be critical in detecting unconventional patterns that might signify security threats.

PC3, with its emphasis on response-related features such as resp_bytes, missed_bytes, and resp_ip_bytes, is particularly revealing. The negative contributions of these features suggest a focus on aspects of network traffic related to response sizes and potentially lost or incomplete data transmissions. This could be indicative of network inefficiencies, errors, or more critically, cybersecurity issues like data interception or tampering.

B. Flawed Data Set

Upon completion of the data set analysis, we have determined that the data set itself may be flawed. Because the information was gathered and tested in a lab, it is likely that the data set did not contain as much noise as randomly gathered network traffic from a real world environment would have. Real world network traffic contains a huge variety of connection types which would add diversity to the features of the data set, but there is no guarantee that malicious traffic will be captured at any given time in the real world thus a lab-generated data set that is guaranteed to contain malicious traffic was used.

The concern with the data set is that the malicious data might be too distinct from benign traffic and that it may lead to overfitting. Our model could become adept at identifying these manufactured differences, which are not representative of real attacks, thus diminishing their effectiveness in practical applications. This situation could give a false sense of security, as the model might perform exceptionally well on the data set but fail to generalize to real-world data, where malicious traffic is not as overt.

V. Discussion, Conclusion, and Future Work

A. Future Work

Future work might encompass a range of activities aimed at enhancing the effectiveness and applicability of our models in network security, as well as the conduction of a deeper analysis of cases where network connections were misclassified. This would aid with understandstanding the limitations of our current models and work towards improving their predictive accuracy.

Understanding why a connection is classified as malicious is as important as the classification itself. Therefore, future work could focus on improving the interpretability of the models, potentially by implementing explainable AI methods. An investigation into whether creating new features or transforming existing ones could enhance model performance, especially for those models that did not perform as well in our initial tests.

Testing the robustness of our models is another key area of focus. Potentially challenging them against adversarial examples or data with simulated attack patterns not present in the initial training data set. This will help assess the models' robustness and their ability to adapt to new and evolving cyber threats.

Handling imbalanced data is a common challenge in cybersecurity. Future research might explore techniques specialized for dealing with such data to improve the accuracy and reliability of our models. To ensure the generalizability of our model, extending analysis to include a broader set of IoT devices and a more varied range of attack types is also essential. By incorporating data from a diverse array of IoT devices and encompassing a wider spectrum of cyberattack scenarios, we can significantly enhance the model's applicability and robustness across different contexts. This expansion will not only provide a more comprehensive understanding of network behavior across various devices but also equip the model to recognize and respond to a broader range of security threats.

Additionally, staying on top of the latest developments in cyberattack methods is crucial. Regular updates and retraining

of the model with current and emerging threat data would be a key focus. This approach ensures that the model remains effective and relevant in the face of rapidly evolving cyber threats.

Moreover, there is an opportunity to delve into the integration of advanced machine learning techniques, such as deep learning and reinforcement learning, which may offer improved detection capabilities, especially in complex and dynamic network environments.

B. Conclusion

In conclusion, our study on network malware detection using machine learning techniques has yielded significant insights and promising results. We successfully analyzed network traffic data between various Internet of Things (IoT) devices, distinguishing between malicious attacks and normal network activities. Our approach demonstrated the viability of using machine learning, particularly methods like the Random Forest model, to reliably detect malicious network traffic using easily accessible network data. This research is particularly relevant given the ever-increasing volume of data transmitted across the internet, largely driven by the proliferation of IoT devices.

Our findings underscore the importance and effectiveness of machine learning in cybersecurity. The high accuracy of our models, especially the Random Forest, in predicting malicious activities in network traffic, points to the potential of these technologies in enhancing network security. The use of Principal Component Analysis (PCA) to reduce the complexity and dimensionality of our data set further highlights the value of sophisticated statistical techniques in making sense of large volumes of data.

The study's relevance is further emphasized by the growing number of cyberattacks involving IoT devices, making the need for robust and efficient detection systems more critical than ever. Our research not only contributes to the academic understanding of network security but also offers practical insights and tools that can be applied in real-world scenarios to protect sensitive data and infrastructure.

As we look to the future, our research opens several avenues for further investigation and development. This includes deeper analysis of misclassified cases, real-time detection implementation, model interpretability, and robustness testing. Additionally, exploring new methodologies and expanding the data set will be crucial in enhancing the models' performance and applicability.

Overall, our study represents a significant step forward in the application of machine learning in the field of cybersecurity. It offers a solid foundation for future research and development in this critical area, paving the way for more secure and resilient network environments in an increasingly connected world.

VI. Contribution of Team Members

A. Jacob Hawkins

NetID: jhawki41
Responsibilities:

- Hyperparameter testing
- Developed visualizations

B. Dylan Lewis

NetID: dlewis37
Responsibilities:

- Model creation
- Model testing

C. Riley Crockett

NetID: rcrocke6
Responsibilities:

- PCA data visualizations and analysis
- Developed visualizations
- Future Work and Conclusion

References

[1] Stratosphere Laboratory. A labeled dataset with malicious and benign IoT network traffic. January 22th. Agustin Parmisano, Sebastian Garcia, Maria Jose Erquiaga. https://www.stratosphereips.org/datasets-iot23

[2] A. Kusar, "IoT Security: 5 cyber-attacks caused by IoT security vulnerabilities", *cm-alliance.com*, Oct. 25, 2022. [Online]. Avalilable: https://www.cm-alliance.com/cybersecurity-blog/iot-security-5-cyber-attacks-caused-by-iot-security-vulnerabilities. [Accessed Dec. 6, 2023]

[3] IBM, "What is random forest?," *imb.com*. [Online]. Available: https://www.ibm.com/topics/random-forest. [Accessed Dec. 7, 2023]

[4] C. Schuman. COSC 425. Class Lecture, Topic: "Ensemble Methods." Dept. of Electrical Eng and Computer Science, University of Tennessee, Knoxville, TN.