Protecting IoT Devices through Localized Detection of BGP Hijacks for Individual Things

DongInn Kim
Indiana University

Vafa Andalibi
Indiana University

Jean Camp
Indiana University

Abstract—**In this paper, we leverage the limited functionality of IoT devices and the homophily of a single home network to identify control plane attacks. We illustrate the use of privacy-preserving data analysis in machine learning to evaluate the leptokurtic distributions of routes from a single device in an individual home in a specific geographic location. Previously, route hijacking has been approached as a large-scale systems problem, requiring network service providers to take action. Route information from the edge has traditionally been considered inactionable, however, small enterprises and homeowners may be targeted for such attacks for reasons ranging from nations attacking suppliers in critical systems to simple monetization of e-crime. We describe how a single small entity can leverage large-scale historical data with their individual histories to identify these attacks. We implement our proposed method in the form of a local agent that monitors the IoT devices and services for detecting BGP hijacking as well as an agent server that utilizes global history in initializing the local agents.**

I. Introduction

The Internet is designed to be agnostic about the routes a packet takes from source to destination. Yet organizations, nations, and even individuals are not so copacetic about rerouting in daily operations. Attackers, seeking to obtain information between remote points of interest, use route hijacking to manipulate traffic so that remote interactions become locally available and immediately accessible. Identification of these hijack events is a challenge; these are usually identified through the network anomalies in the data plane, the control plane, or a combination of these with a resulting ”yes or no” binary answer from a highly expert human decision-maker.

The challenge of identification of control plane attacks on the edge is made more difficult by the lack of readily available global routing data as well as the lack of expert personnel to evaluate these. We approach this challenge by combining global routing information and local histories (including routing history and geographical information). Essentially we leverage the homophily of each device. Each individual thing is relatively simple and stationary resulting in a high level of homophily and a limited number of remote connections. This homophily can be used to identify anomalies from the edge. One goal in our design is to enable a non-expert human decision-maker to apply their own complex political, economic, and deeply contextual risk assessments to choose between availability (i.e. send it now) and confidentiality (i.e. delay sending until the route is trusted). To identify anomalies we use cosine similarity and personalize the likelihood that a route from and to a specific endpoint is legitimate. We include data about the physical geographical layer and leverage local historical data, in addition to big data.

For those not familiar with control plane attacks, a route identifies the set of hops that a packet should traverse across the autonomous systems (AS) to its destination. Each hop between route and destination is selected using the Open Shortest Path First (OSPF) algorithm and the resulting path is distributed via the Border Gateway Protocol (BGP) routing protocol. This is constantly changing as routes are updated. It is not feasible to simply reject or ignore an update, as this would have disruptive implications for the routing information base (RIB) as a whole. The vast majority of these updates are honest and benign. BGP anomalies arise in the autonomous routing environment when ASes make inaccurate claims about association or ownership, which may be unintentionally disruptive. BGP hijacking occurs when maliciously forged routing announcements are sent with the goal of perturbing traffic.

Potentially harmful BGP anomalies occur due to route hijacking, misconfiguration, and Denial of Service (DoS) attacks. This paper focuses on the mitigation of the BGP hijacking only from the client's point of view by utilizing the public BGP stream dump datasets and ML models. A challenge of detecting BGP hijacking is not just to defeat hijacks, but also to verify their initiation and confirm that an anomaly is a hijack rather than a response to congestion or other normal operation. We illustrate that the use of only five features alone can identify some hijacks at the edge.

We ground the detection of hijacks in an analysis of cosine distance using only five features to evaluate the efficacy of an explainable detection mechanism: country, bordering countries, continent, neighbor continents, and neighbor AS. These observable features were also chosen because they support human decision-making; that is, one goal of our work is to create a ML approach that complements human intelligence. We report a high level of efficacy for the identification of hijacks using this human-centered design. We compare five different ML approaches and find that the additional opacity of random forest, decision tree, SVM, logistic regression, and KNN does not have a corresponding payoff in terms of performance even with unbalanced datasets.

The contribution of this paper is to illustrate an innovative approach to detecting BGP hijacking at the endpoint, identifying anomalies by leveraging a combination of highly homophilous local data and global data. We present a proof of concept that identifies the IoT services and IoT devices by utilizing the cosine similarity and geolocation of each AS to its neighboring ASes from the BGP stream dump data and illustrate the implemented local agent which determines the AS routes of IoT devices and edge servers. We illustrate the efficacy of this approach and argue that a highly distributed method such as the one proposed here would be a complement to detection approaches in the center of the network.

II. Motivation

BGP is the protocol used to define the route of packets between the autonomous networks (AS) which comprise the Internet. Hijacks occur when an AS misrepresents its ownership or its place in the network topography [1]. The research on detecting and classifying BGP hijacks began with the first public BGP hijacking, April 1997. Beyond this malicious activity, other incidents have been due to typos, link failure, and misconfiguration of BGP. It can be difficult to distinguish the differences between legitimate changes and malicious activities.

While some local methods for protecting the IoT devices in end-users' networks have been suggested to prevent the network hijacking [2], [3], hijack detection, mitigation, and prevention methods have been primarily managed by the network operators, not the end-user. Network service providers have access to immense amounts of routing data, which can be updated and aggregated to inform route decisions. Network systems personnel are highly skilled and focused on operations. Multiple analytical techniques have been used: Fast Fourier Transform (FFT) [4], Daubchies5 (db5) Wavelet transform [5], Recurrence Quantification Analysis (RQA) [6], as well as other ML models [1]. One challenge to these ML approaches is the correct labeling of routes [7], [8]. Our localized data approach leverages data that can be labeled with a high degree of confidence.

Cryptographic preventative technologies have been deployed to modify the BGP mechanism by authenticating BGP updates. Resource Public Key Infrastructure (RPKI) [9] would prevent the false announcement of smaller components (e.g. $/26s$ in IPv4) as used in route hijacking. Recently Amazon announced deployment of RPKI within its systems. In comparison with Amazon's RPKI system, the BGPSec [10] system does not require any action by the network operator. It also offers the ability for individuals to set their risk thresholds through a level selection for alerts.

A current organizational approach is the Mutually Agreed Norms for Routing Security [11] (MANRS), which requires managerial commitments from the operators of the ASes. The adoption of this has varied widely, as shown by the regularly updated data and geographical visualizations available at the MANRS Observatory.

Authenticating solutions, including RPKI, ROV, ROV++, and BGPSec, address different types of attacks yet have three critical similarities relevant to this work: there must be a minimal level of adoption in the network for these to be effective; the network operator providing service to the individual IoT thing must adopt these; thus they are beyond the control (or even the knowledge) of the individual at the edge.

Most approaches to the detection of BGP hijacking are performed using control plane data [1] which is responsible for determining the routing path in the routing table. Our approach similarly uses the control plane but differs in the method of detecting the anomalies in the BGP prefix. It is based on the localized data as well as all the possible routing anomalies. Historical global BGP Autonomous System Numbers (ASN) routing data is publicly available and provides valuable information about the adjacency of ASes. Directly connected ASes are referred to as neighbors. The frequency of accessing a neighbor AS indicates the ranking of AS neighbors from most to least likely. This global information is combined with the location of an installed IoT device, its default network, and its patterns of connections. Our local agent monitors the IoT devices and associated services by mapping the likelihood of paths through the network. The localized routing data that we collected from the local agent and the agent server are considered as the routing history between the IoT devices and their services.

The agent server was also developed to utilize local data reported by the connected local agents for initializing the new local agents when additional devices are installed (otherwise every device needs to initiate all the local history of the network traffic on installation) [2]. We leverage both this data and global data to periodically update the measures of AS adjacency for multiple purposes. First to verify that there is not local drift or subversion of the agent; that is, to ensure that the agent has not accepted a hijack as valid over the long. Second, to avoid false positives based on unexpected network events. Such evaluation is needed for large network perturbations. For example, recently the bombing of Nashville on Christmas 2020 destroyed an AT&T switching center causing immediate significant changes in network adjacency. This would not be locally observable, and blocking connections as malicious could have occurred without a lack of global updates. Such local false positives could have exacerbated the outages caused by this event.

Previous research has integrated data from both the data plane and the control plane to improve accuracy and efficacy in detecting the BGP hijacks [12], [13], [14]. Building on this previous work we similarly leverage identifying factors including IP addresses, round trip time (RTT), number of hops, and packet sizes which can help to determine the specific traffic or routes in the data plane point of view. These data are also useful to identify specific IoT devices and their services on the network [15], [16], [17].

We begin with an unusual hypothesis that it is possible and reasonable to expect a local agent to be able to detect a Man in The Middle (MITM) or route hijacking attack despite the fact that these anomalies are remote occurrences that may appear to be normal routing events [2]. To test this we implemented a proof of concept in the form of a local agent that integrates the analysis of the BGP dump data with local information. The BGP hijacking data was collected from the Cisco BGPStream team [18]. We experimented with cosine similarity and geolocation vector distance to test how accurately such data can be leveraged to detect the hijacks. The cosine similarity and geolocation vector distance define the similarity of the neighboring ASes and allow us to build the map of the most common and probable AS routes for an ASN.

Historical control plane data can be downloaded from the Route Views project [19] from CAIDA and Réseaux IP Européens (RIPE) [20]. In our initial selection of features, we examined BGP data using different attacks that are available from different entities. In the design of this system, we have leveraged data categorized into Multiple Origin AS (MOAS), forged path, and Black-holing. Researchers use the NANOG mailing list [21], Spamhaus Don't Route Or Peer (DROP) ASN list [22], as well as locally obtained data plane information (e.g. TTL, Number of Hops, and RTT).

The most common BGP attributes used for the hijack detection are Origin, AS Path, LOCAL-PREF, AGGREGATOR, and Multi Exit Discriminator (MED). In this study, we focus on the AS Path. One of the approaches to detect the BGP hijack is based on hijackers' behavioral patterns. They applied a ML model to automatically identify ASes with BGP origination patterns similar to serial hijackers [23]. Identifying a valid MOAS from an attack is difficult, and in this study, we will use the AS similarity and geolocation distance to propose a method for valid hijack detection.

III. Method

The goal is to develop a systematic verification solution to detect the hijacked ASN routes by applying the cosine similarity of the target features on each AS node and calculating the geolocation distance between AS nodes. This is to verify the BGP anomalies between the IoT devices and their edge server. The network traffic between the IoT devices and the edge servers is collected in the local agent which provides the wireless access point for the IoT devices. The AS routes between the IoT devices and the edge servers are consistent; a few number of variations can be found in Table I. The AS routes collected from the local agent are persistent even for different locations with the same Internet Service Provider (ISP) as shown in Table II, but the number of repetitions of an AS can vary. The anomalies that the local agent found were simulated with the local MITM attack and they can be detected immediately as displayed in Table III.

We have run the analysis of the BGP routing patterns based on their cosine similarity and geolocation dependency. The found pattern will be used to verify the justification of the anomalies in the IoT traffic.

Note that the local agent uses deep packet inspection, traceroute, and BGPStream data on those packets between the IoT devices and the corresponding edge servers. It uses these to determine the expected routing behaviors for each edge service. The local AS route data is part of the BGPStream dump data. The machine learning model is trained with the BGPStream dump data and Cisco BGP hijacking data. Deviations from expected behavior are identified by the local agent using this global data and verified using cosine similarity.

For example, the CAIDA [24] dump data can not verify the above anomaly because the local AS numbers (AS198949 and AS0) are not registered or stored in the dump data. In contrast, the local agent is designed to detect any anomaly in the local routes, and therefore missing the BGP hijacking verification is not a risk for the system. In general, the verification system that we developed provides the current possible ASN routes based on the given source and target ASes. As presented in Table IV, the actual AS route is highlighted from the source AS `[87 (1:US)]` and target AS `[15169 (4:US)]`. The predicted routes can be narrowed down based on the number of hops. The 7 routes in Table IV are identified using only 4 hops. If we increase the number of hops to 5, the total routes would be 98. The number of total possible routes can be even bigger if we do not restrict the possible geolocation. The 98 routes are only for the United States. If we allow the non-USA routes to be in the possible intermediate route, the number of routes can be very large (341). The local agent can utilize our verification system to make sure that the unexpected anomaly in the public AS is valid or not. If the number of hops is increased more than 5, the possible routes would exponentially grow due to the routing history.

This analysis assumes that the property of each AS node

TABLE I: Routes from Google Home (located at Indiana University) to the Edge server (HTTPS) with 74.125.124.188.

	Indiana University		
Source IP	149.165.234.130	149.160.244.237	149.160.204.149
ASN Routes	[AS198949]	[AS198949]	[AS198949]
	[AS198949]	[AS87]	[AS87]
	[AS198949]	[AS87]	[AS87]
	[AS19782]	[AS19782]	[AS19782]
	[AS19782]	[AS19782]	[AS19782]
	[AS19782]	[AS19782]	[AS19782]
	[AS19782]	[AS19782]	[AS19782]
	[AS2381]	[AS2381]	[AS2381]
	[AS15169]	[AS15169]	[AS15169]
	[AS15169]	[AS15169]	[AS15169]
	[AS15169]	[AS15169]	[AS15169]
	[AS15169]	[AS15169]	[AS15169]
Destination IP	74.125.124.188		

TABLE II: Routes from Google Home (located at a Comcast home network) to the Edge server (HTTPS) with 74.125.124.188 showing both self-similarity and slight differences in routing compared with Indiana University above.

	Comcast		
Source IP	68.50.16.40	68.51.124.139	50.195.244.27
ASN Routes	[AS198949]	[AS0]	[AS7922]
	[AS7922]	[AS7922]	[AS7922]
	[AS7922]	[AS7922]	[AS7922]
	[AS7922]	[AS7922]	[AS7922]
	[AS7725]	[AS7725]	[AS7725]
	[AS7922]	[AS7922]	[AS7922]
	[AS7922]	[AS7922]	[AS7922]
	[AS7922]	[AS7922]	[AS7922]
	[AS7922]	[AS7922]	[AS7922]
	[AS7922]	[AS7922]	[AS7922]
	[AS15169]	[AS15169]	[AS15169]
	[AS15169]	[AS15169]	[AS15169]
	[AS15169]	[AS15169]	[AS15169]
	[AS15169]	[AS15169]	[AS15169]
	[AS15169]	[AS15169]	[AS15169]
Destination IP	74.125.124.188		

TABLE III: A MiTM attack is applied to the local agent. A new ASN [AS0] is added to the existing ASN route

Google Home	
Normal	After MiTM
[AS198949]	[AS198949]
[AS87]	**[AS0]**
[AS87]	[AS87]
[AS19782]	[AS87]
[AS19782]	[AS19782]
[AS19782]	[AS19782]
[AS19782]	[AS19782]
[AS2381]	[AS19782]
[AS15169]	[AS2381]
[AS15169]	[AS15169]
[AS15169]	[AS15169]
[AS15169]	[AS15169]
[AS15169]	[AS15169]
	[AS15169]

TABLE IV: Predicted possible ASN routes with the given source and target ASN.

Possible ASN routes			
Source: Indiana University			
[87 (1:US)]	[26415 (2:US)]	[2381 (3:US)]	[15169 (4:US)]
[87 (1:US)]	[26415 (2:US)]	[3491 (3:US)]	[15169 (4:US)]
[87 (1:US)]	[26415 (2:US)]	[6461 (3:US)]	[15169 (4:US)]
[87 (1:US)]	[26415 (2:US)]	[6939 (3:US)]	[15169 (4:US)]
[87 (1:US)]	[19782 (2:US)]	[11164 (3:US)]	[15169 (4:US)]
[87 (1:US)]	**[19782 (2:US)]**	**[2381 (3:US)]**	**[15169 (4:US)]**
[87 (1:US)]	[19782 (2:US)]	[6939 (3:US)]	[15169 (4:US)]
Destination IP: 74.125.124.188			

depends on the CAIDA dump data that RIPE RIS and RouteViews provide, the collected geolocation data covers 89%, i.e., $(1 - 7,497/67,928) \times 100$, of all unique ASes, and cosine similarity of each node is precalculated.

The RIPE RIS dump data files from a specific date (e.g. 04/29/2020) have all unique 7,691,630 AS Paths and 67,931 ASNs. The location of each AS has been collected for all 67,928 ASes and we could not identify the geolocation of 7,497 ASes due to various reasons, e.g. the network service provider is no longer in business or route flapping between disparate locations. We assigned a country code to each ASN by referring to its BGP routing information. The experimental data was generated by extracting all the necessary features from the CAIDA dump data and then retrieving their geolocations. Next, the ASNs are mapped to their neighbor ASes and the results are saved. The file containing the neighbor mapping can be loaded to the memory so that the neighbors of an ASN can be quickly retrieved when cosine similarity to the next ASN in a route is being calculated.

Data Source and Workflow

1) Data collection: The BGP data we used here is collected from RIPE RIS and RouteViews (from April 29, 2020). The CAIDA data have two sets of files: the first set of files (prefixed by ‘updates’) is created every 5 minutes for all BGP packets, the second set (prefixed by ‘bview’) is created with the entire BGP routing table every 8 hours. We used the second set to map all the neighbor ASes to the corresponding ASN at the ‘Data collection’ step in Fig 1. The RouteViews data provides Routing Information Base (RIB) and updates collected by 26 collectors.

We also collected specific BGP data from the local agent deployed in several locations. The location with the most data had the following devices on its network:

- Google Home: a smart hub to communicate through the WiFi. The access domain ranges are managed with googol addresses (*.1e100.net). We used the routing data from Google Home as shown in Table I, II, III, and IV.
- Ring Doorbell: a camera-enabled smart doorbell. It is connected to the AWS edge servers to record, store, and retrieve the video from the doorbell camera.
- WeMO Smart Plug: a Belkin Smart Plug that uses the Open WRT firmware to control the devices and communicate with its edge server. WeMO App is used to control the smart plug through the smartphone.
- Philips Hue: a Zigbee hub to communicate with the Zigbee enabled IoT devices, such as Philips bulbs. The Hue device can be accessed through the Philips App in the same WiFi network.
- Amcrest IP camera: an IP camera with the motion-detection feature. Amcrest App can be used to control the camera.

In addition, we collected and analyzed data from each location. We observed that these routes are more likely to be leptokurtic, meaning that the data are closely concentrated around the mean and also that the tails, while small in magnitude, are long. In other words, the routes from IoT devices are likely to be highly similar (i.e., concentrated around the mean) but will have excess positive kurtosis (i.e., extreme events may happen). In our analysis, we are particularly concerned with those extreme events. As a result, cosine distance should be a feasible approach as there may be many small deviations (for example from content delivery networks) while the attacks using route hijacking will result in very few large deviations.

2) Data extraction: We used the `bgpreader` command line tool available in the `libBGPStream` library (developed by BGPStream[1]) to parse and extract the useful features (e.g. Origin AS, AS-path, and prefix) from the dump data (e.g. `rib.20200429.0000.bz2`, `updates.20200429.0000.gz`). The same features can be extracted using `bgpdump`, which is available in the `libBGPDump` library. The extracted features are used to map the neighbor ASNs to a given ASN and this mapping lets us generate the possible ASN routes. The extracted AS-path is sorted to find the unique AS-paths and unique ASNs.

3) Pre-processing: We identified all AS adjacencies through an examination of the data with ASNs representing nodes and every node with at least one edge labeled as a feasible path. Please note we attempt to map actual autonomous systems then operate off the assigned numbers. The node-by-node mapping enables a list of all feasible routes based on a history of the interactions of the numbered ASes. Using this we then created an exhaustive map of every ASN route from a subset of ASNs. These ASNs were chosen, as described

[1] https://bgpstream.caida.org/

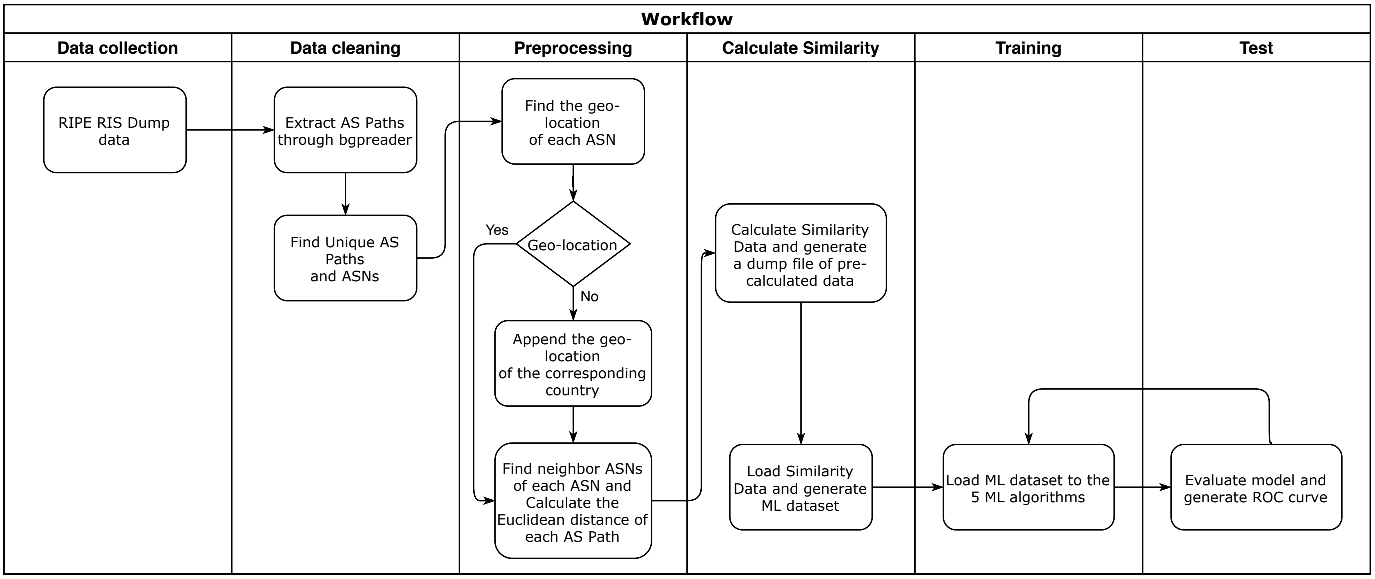

Workflow
Data collection
Data cleaning
Preprocessing
Calculate Similarity
Training
Test
RIPE RIS Dump data
Extract AS Paths through bgpreader
Find Unique AS Paths and ASNs
Find the geo-location of each ASN
Yes
Geo-location
No
Append the geo-location of the corresponding country
Find neighbor ASNs of each ASN and Calculate the Euclidean distance of each AS Path
Calculate Similarity Data and generate a dump file of pre-calculated data
Load Similarity Data and generate ML dataset
Load ML dataset to the 5 ML algorithms
Evaluate model and generate ROC curve

Fig. 1: Workflow of the BGP hijacking analysis

The flow chart of the detection model presented here, beginning with global data compilation and processing, including local thing-specific data observations, and ending with results.

below, based on accessibility for data compilation. This step is required upon initiation and can be updated offline.

A straight-forward python script generated this CSV dataset with unique ASNs containing their neighbor ASes, and this dataset was augmented with geolocation and jurisdictional data. The neighbor ASes are populated from the full mapping data by simply loading the appropriate data file. The geolocation of each AS is obtained by looking up the geolocation of the hosting IP address in the AS. Note that some ASes have registered their domains in multiple places, therefore, it is not possible to populate their geolocation values with only a single location. In this case, we chose the geolocation of the main domain. The generated CSV dataset has 67,928 records and 7,497 records that do not have geolocation. It is about 89% of the total dataset. For those 7,497 records, we used the geolocation of the country where the ASN resides in. At this step, we also calculated the Euclidean distance between ASNs that are in an AS-path.

4) Calculate Similarity: The cosine similarity of an ASN is useful to find the most frequent neighbor ASNs from similar ASNs. The 5 attributes for the cosine similarity are *country*, *neighbor countries*, *continent*, *neighbor continents*, and *neighbor ASNs*. The biggest number of neighbor ASNs is 7,957 and the average number of neighbor ASNs is about 4.90. The 5 attributes are to verify the probability of a route from a certain ASN to a certain neighbor of that ASN. The objective is that for any ASN, we can find which neighbor ASN is the most probable next hop. At first, we find similar ASNs to the initial ASN by using the cosine similarity (AS-path is used as a path vector). Each similar ASN knows its neighbor ASNs and the frequency of visiting them. Based on the visiting frequency we can rank the next ASN hops. The calculation of the cosine similarity with the big dataset (67,931 unique ASNs, 5 attributes on each ASN, and various numbers of neighbor ASNs in the neighbor ASN attribute) is expensive and it is not necessary to have this calculation in real-time. We save the precalculated similarity data and then use it for generating the ML dataset which contains the following features:
class, old_asn, next_asn, similarity, distance, old_country, next_country, neighbor_countries, hijack_distance, source_asn, target_asn, max_distance, min_distance, total_distance, path, hijacked_asns.

5) Training: We have deployed the same ML dataset to five supervised ML algorithms: *Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), K-Nearest Neighbor (KNN),* and *Support Vector Machine (SVM)*. The hijack 'class' of the dataset was determined by the Cisco group in 'bgp-stream.com'. The benign class was extracted from the BGP data where the AS-Path does not belong to the aforementioned hijack dataset. Since the dataset is imbalanced due to the small number of hijack dataset (287 hijacks out of 5,195 total data), we have implemented two experiments for the ML evaluation: first, using the imbalanced dataset, and second, using the adjusted dataset, anticipating that the cosine similarity value and the geolocation distance value would be important features in classification. The ML evaluation is discussed in section IV.

6) Test: The 10-fold cross-validation was applied to the five algorithms mentioned in the previous section (III-5). The ML models are re-trained and then the new classification results are calculated.

For ground truth, we used a labeled Cisco BGP hijacking dataset and labeled our own hijack. We examined deviations to verify the anomalies identified for our home participants. Other

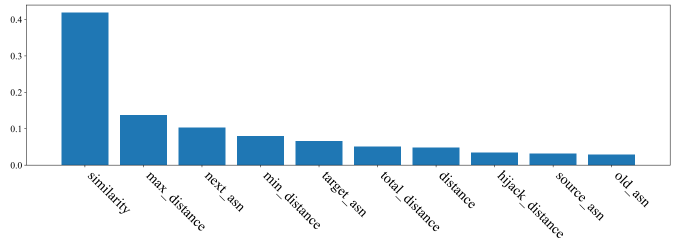


Fig. 2: Feature importance on the Decision Tree and Random Forest model showing that the same factors dominate in both approaches.

TABLE V: Performance of the applied ML models with a fully imbalanced dataset.

	Machine Learning Models				
Name	RF	DT	LR	KNN	SVM
Eval Time [s]	0.637	**0.116**	1.034	0.273	27.205
Accuracy	**0.984**	0.966	0.962	0.957	0.945
Precision	0.976	**0.979**	0.894	0.876	0.473
Recall	**0.878**	0.693	0.698	0.647	0.500

than that which is labeled we assume that the BGPStream dump data are benign.

IV. Results and Discussion

We used five features for the cosine similarity value in our experiment: *country, neighbor countries, continent, neighbor continents,* and *neighbor ASes*. The similarity of the word counts in the five features is vectorized. We report on results using the top 5 similar ASNs; meaning we ranked their neighbors based on the number of times the neighbors were accessed by the top 5 ASes. Our cosine similarity distance is calculated based on the rank of the similarity value. As shown in Fig 2, the most important feature to classification is similarity and the second most important feature is the geolocation feature (e.g, max_distance, min_distance, and total_distance).

The geolocation (latitude and longitude) value is used to calculate the Euclidean distance between ASNs. For the 7,497 ASNs that are missing geolocation values, we instead used their country code for calculating the routing distance. With these two important features, we determine which ML model has the best performance (e.g. evaluation time, accuracy, precision, and recall) and use this to verify the anomalies that our local agent found.

As shown in Table V and VI, Random Forest (RF) outperforms other models in terms of accuracy (98.4%). With the second-highest accuracy of 96.6%, Decision Tree (DT) has the shortest evaluation time of 0.119 seconds. The featured dataset does not fit well with Support Vector Machine (SVM) as SVM is designed to weigh on instances and not on features. SVM shows very poor performance on our imbalanced dataset and applying it on such dataset takes a disproportionate time (31.452 seconds). Logistic Regression (LR) is usually used to work with a hyperplane that separates the feature space into classes. The poor accuracy of LR is due to a large number of features in the dataset (16 features) which cannot be divided into classes by the hyperplane of the LR. LR ended up having poor precision and recall too as it could not find the local minimum with L-BFGS[2].

TABLE VI: Performance of the applied ML models with the adjusted dataset.

	Machine Learning Models				
Name	RF	DT	LR	KNN	SVM
Eval Time [s]	0.630	**0.119**	1.243	0.270	31.452
Accuracy	**0.984**	0.966	0.962	0.956	0.945
Precision	0.964	**0.977**	0.887	0.870	0.473
Recall	**0.857**	0.693	0.698	0.647	0.500

Regardless of the value of the k, K-Nearest Neighbor (KNN) classifier showed poor precision (0.876) and recall (0.647). The relatively short evaluation time of this approach (0.270 seconds) is due to the small size of the dataset and will grow as the dataset gets bigger. Note that KNN does not directly learn a discriminative function from the training data but keeps the training dataset in the memory instead. It compares the distances between the k neighbors of the new data and classifies it based on the majority of its k neighbors.

In RF, we got the best results with 10 trees in the forest and measured the feature importance as the averaged impurity decrease computed over all decision trees in the forest. The feature importance is generated without any assumption about whether our data is linearly separable or not. Since RF works with multiple trees and aggregates their results, the evaluation time RF must be slower than DT. Random Forest creates the best performance in our Receiver Operating Characteristic (ROC) experiment as shown in Fig 3.

As shown in Fig 3 (b), adjusting the dataset did not considerably improve the accuracy of the 5 ML algorithms (RF, DT, LR, KNN, and SVM). Hence, we expect the trained model on RF and DT to consistently work with the real-world dataset (which is also imbalanced).

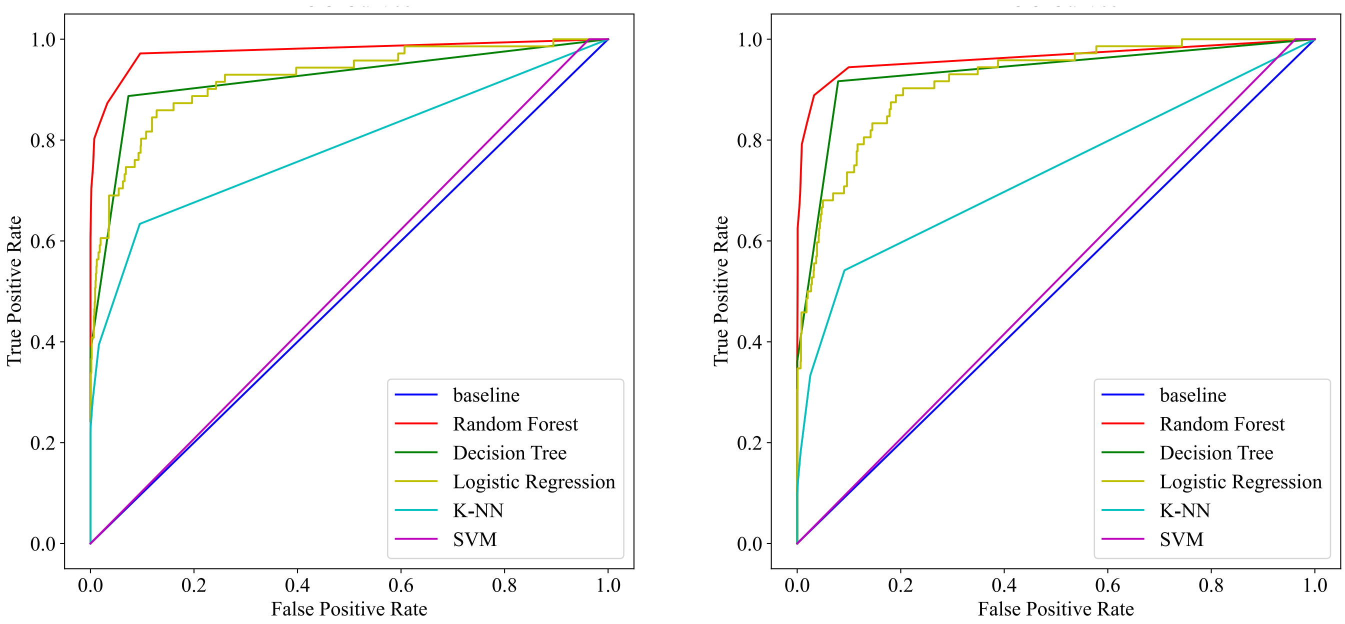
We found that most hijacking ASNs have a certain similarity value greater than 10. The bigger similarity value means less similarity to the initial ASN. The majority of benign data has a similarity value below 10. A further analysis step would be to filter out the data with values of less than 10, creating a smaller dataset of benign and hijacked data with similarity values of greater than 10. We would expect the distance features to be more important for the classification of such a dataset. Meanwhile, the distance features can be used to verify the decision of the current ML algorithms.

Accordingly, our local agent uses the RF models to acquire the highest accuracy in detecting the BGP hijacking by utilizing the most likely neighbor ASes. The neighboring AS map is to be used for verifying the given anomaly's validity.

V. Conclusions and Future Work

In this study, we presented a ML-based approach to detect BGP hijacking as a verification system of routing anomaly using a combination of BGP history, geographical stability, and the homophilous network behavior of both human households

[2] L-BFGS: Limited-memory Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm is used to find the local minimum of an objective function [25]



(a) Unaltered Dataset

(b) Adjusted Dataset

Fig. 3: ROC Curves of 5 different models applied on (a) unaltered dataset and (b) adjusted dataset

and their specific IoT devices. We used cosine similarity based on five features and illustrated its efficacy in distinguishing hijacking from benign anomalies. In terms of efficacy of detection, we used five supervised ML algorithms to verify the validity of our approach. We found Random Forest and Decision Tree to be superior algorithms for this purpose while the performance of SVM model suffered from the imbalanced dataset and had the worst performance. Attempts on adjusting the dataset manually did not affect the performance of the algorithms.

Our future work includes the installation of this device into homes with non-expert residents to determine if they find the detection and mitigation of the risk of hijacks usable and acceptable. Our long term goal is to leverage offline human-scale data as well as network scale big data to support human decision-making, creating a highly understandable approach to detect BGP hijacks. We also intend to fine tune the particular thresholds of the cosine similarity and distance to generate the better classification. In addition, determining how often to randomly evaluate cosine similarity to avoid consistent false positives is a component of future work. Finally, we posit that the efficacy of this approach is correlated with the simplicity of the device and thus its likelihood of being unable to support an end-to-end encryption. Such a correlation would indicate that this approach is appropriate for the most simple, and potentially vulnerable, IoT devices. This hypothesis will be evaluated with the multi-residence deployment planned for late in 2021.

Acknowledgments

This research was supported in part by the National Science Foundation awards CNS 1565375 and CNS 1814518, the grant #H8230-19-1-0310, Cisco Research Support, Google Research, and the Comcast Innovation Fund. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the US Government, the National Science Foundation, the National Security Agency, Cisco, Comcast, Google, nor Indiana University.

References

[1] B. Al-Musawi, P. Branch, and G. Armitage. Bgp anomaly detection techniques: A survey. *IEEE Communications Surveys Tutorials*, 19(1):377–396, 2017.

[2] DongInn Kim, Vafa Andalibi, and L Jean Camp. Fingerprinting edge and cloud services in iot. In *2020 13th International Conference on Systematic Approaches to Digital Forensic Engineering (SADFE)*, pages 13–21. IEEE, 2020.

[3] Vafa Andalibi, DongInn Kim, and L Jean Camp. Throwing mud into the fog: Defending iot and fog by expanding mud to fog network. In *2nd USENIX Workshop on Hot Topics in Edge Computing (HotEdge 19)*, 2019.

[4] Peter Bloomfield. *Fourier Analysis of Time Series: An Introduction*. Wiley, 1976, November 2007.

[5] Yinglian Xie, Hyang-Ah Kim, David R. O'Hallaron, Michael K. Reiter, and Hui Zhang. Seurat: A pointillist approach to anomaly detection. In Erland Jonsson, Alfonso Valdes, and Magnus Almgren, editors, *Recent Advances in Intrusion Detection*, pages 238–257, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[6] Norbert Marwan, M. Carmen Romano, Marco Thiel, and Jürgen Kurths. Recurrence plots for the analysis of complex systems. *Physics Reports*, 438(5):237 – 329, 2007.

[7] Ricardo Bennesby da Silva et al. Deepbgp: A machine learning solution to reduce bgp routing convergence time by fine-tuning mrai. 2019.

[8] Qingye Ding, Zhida Li, Prerna Batta, and Ljiljana Trajković. Detecting bgp anomalies using machine learning techniques. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 003352–003355. IEEE, 2016.

[9] Josh Bailey, Dean Pemberton, Andy Linton, Cristel Pelsser, and Randy Bush. Enforcing rpki-based routing policy on the data plane at an internet exchange. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, page 211–212, New York, NY, USA, 2014. Association for Computing Machinery.

[10] Q. Li, J. Liu, Y. Hu, M. Xu, and J. Wu. Bgp with bgpsec: Attacks and countermeasures. *IEEE Network*, 33(4):194–200, 2019.

[11] Manrs. Mutually Agreed Norms for Routing Security. [Accessed on Jan 29 2020], 2020. https://www.manrs.org/.

[12] Xingang Shi, Yang Xiang, Zhiliang Wang, Xia Yin, and Jianping Wu. Detecting prefix hijackings in the internet with argus. In *Proceedings of the 2012 Internet Measurement Conference*, IMC '12, page 15–28, New York, NY, USA, 2012. Association for Computing Machinery.

[13] X. Hu and Z. M. Mao. Accurate real-time identification of ip prefix hijacking. In *2007 IEEE Symposium on Security and Privacy (SP '07)*, pages 3–17, 2007.

[14] Fabian Fischer, Johannes Fuchs, Pierre-Antoine Vervier, Florian Mansmann, and Olivier Thonnard. Vistracer: A visual analytics tool to investigate routing anomalies in traceroutes. 10 2012.

[15] Jaidip Kotak and Yuval Elovici. Iot device identification using deep learning. *Advances in Intelligent Systems and Computing*, page 76–86, Aug 2020.

[16] Ola Salman, Imad Elhajj, Ali Chehab, and Ayman Kayssi. A machine learning based framework for iot device identification and abnormal traffic detection. *Transactions on Emerging Telecommunications Technologies*, 09 2019.

[17] Rajarshi Roy Chowdhury, Sandhya Aneja, Nagender Aneja, and Emeroylariffion Abas. Network traffic analysis based iot device identification. In *Proceedings of the 2020 the 4th International Conference on Big Data and Internet of Things*, BDIOT 2020, page 79–89, New York, NY, USA, 2020. Association for Computing Machinery.

[18] Cisco Systems. Cisco BGPGream. [Accessed on Jan 29 2020], 2020. https://bgpstream.com.

[19] Route Views. RouteViews BGP Data. [Accessed on Jan 29 2020], 2020. http://routeviews.org/.

[20] RIPE RIS. Ripe bgp data. [Accessed on Jan 29 2020], 2020. https://www.ripe.net/analyse/internet-measurements.

[21] NANOG. North america nog. [Accessed on Jan 29 2020], 2020. https://www.nanog.org/.

[22] SPAMHAUS. Spamhaus drop list. [Accessed on Jan 29 2020], 2020. https://www.spamhaus.org/drop/asndrop.txt.

[23] Cecilia Testart, Philipp Richter, Alistair King, Alberto Dainotti, and David Clark. Profiling bgp serial hijackers: Capturing persistent misbehavior in the global routing table. In *Proceedings of the Internet Measurement Conference*, page 420–434, New York, NY, USA, 2019. Association for Computing Machinery.

[24] CAIDA. Caida bgp data. [Accessed on Jan 29 2020], 2020. https://www.caida.org/home/.

[25] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.