Unsupervised One-Class Learning for Anomaly Detection on Home IoT Network Devices

Jonathan White
Computer Science Research Centre
University of the West of England
Bristol, UK
jonathan6.white@uwe.ac.uk

Phil Legg
Computer Science Research Centre
University of the West of England
Bristol, UK
phil.legg@uwe.ac.uk

***Abstract*—In this paper we study anomaly detection methods for home IoT devices. Specifically, we address unsupervised one-class learning methods due to their ability to learn deviations from a single normal class. In a home IoT environment, this consideration is crucial as supervised methods would result in a burden on many non-technical consumers which could hinder their effectiveness. For our study, we develop a home IoT network monitoring tool, and we illustrate network attacks against a variety of typical home IoT devices. As a result, we propose measures that could aid home consumers in defending ever-increasing home IoT networks.**

***Index Terms*—IoT, anomaly detection, one-class learning**

I. Introduction

The proliferation of home Internet of Things (IoT) devices has led to a dynamic change in terms of size and complexity of home networks. Many homes now incorporate a number of "smart" devices, including televisions, kitchen appliances, lighting, power switches, access controls and security monitoring, not only to simplify and automate everyday tasks, but also to improve our quality of life and wellbeing. Interactions with devices can be made using mobile phones, smart voice assistants, or other motion and proximity sensors. However, each new device connected to the home network increases the security risks posed within the network. Whilst device manufacturers are keen to adopt consumer IoT, the rush to production can result in poor product design with a lack of security consideration. From a consumer standpoint, IoT devices often lack the same security needs that consumers may expect from a traditional computing device, such as a desktop PC or a laptop [1], yet IoT devices may contain sensitive personal information such as usernames and passwords. According to the Open Web Application Security Project (OWASP), weak passwords and username enumeration are among its top 10 vulnerabilities [2]. In 2016, the Mirai botnet exploited weak credentials in consumer IoT to launch Distributed Denial of Service (DDoS) attacks. In 2017, BrickerBot malware utilised a similar attack to leverage default SSH credentials and misconfigurations in IoT devices to 'brick' them by corrupting the device firmware, so that the devices are then permanently broken [3]. The smart home market is projected to show an annual growth rate of 18.2% between 2020-2023 resulting in a market volume in 2023 of 139.8 billion USD and a total of 75.44 billion IoT devices worldwide by 2025 [4]. Yet many devices have been reported to contain security flaws due to improper implementation and misconfigurations [5], [6]. Consumers will naturally expect home appliances to be operational for many years after purchase, therefore it is crucial that support and security updates of IoT-enabled devices reflects the true life cycle of consumer ownership.

Whilst home consumers may be familiar with security measures such as anti-virus and firewall applications from their traditional computing experience, they are likely less familiar with how to protect their IoT devices. At an enterprise level, it is common to deploy Intrusion Detection Systems (IDS), which may be host-based (HIDS) or network-based (NIDS). Enterprise IT security solutions usually consist of a combination of static perimeter defences such as firewalls and IDS, coupled with host-based defences such as end-point detection and response (EDR), anti-virus and anti-malware software, and regular patching of software vulnerabilities [7]. Traditional enterprise IT-based IDS may be less efficient and/or inadequate for IoT systems due to issues such as heterogeneity, use cases, constrained resources and connectivity [8]. Such corporate security solutions are unsuitable for deployment in a home network environment due to the complexity of the solution, costs associated with expensive enterprise-grade software and hardware, and the ability of non-technical users to understand the outputs of these tools. Recent attacks have exploited IoT devices as botnets for distributed denial-of-service attacks [9]. Such attacks can be identified using behavioural analysis [10] or anomaly detection using network traffic analysis [11].

Machine learning has become widely used for anomaly detection in a number of security applications, including network traffic analysis [12], [13], malware analysis [14], [15] and insider threat detection [16]. Supervised classification models require training of respective classes, such as 'malicious' and 'benign'. This can pose challenges in the security domain, since we may not have training examples of all 'malicious' possibilities, and the number of 'benign' samples may significantly outweigh the number of 'malicious' samples available. One approach to overcome this is to learn a model of a single class, and use this to model deviations from some underlying 'normal' behaviour. In this paper, we study how this approach,

Funded by the UK National Cyber Security Centre (NCSC).

often referred to as one-class learning, can be utilised to improve IoT device security. As an unsupervised learning method, one-class learning can offer a scalable approach for monitoring large and complex IoT networks, whilst also being flexible enough to handle a variety of different attack patterns exhibited by different devices. Furthermore, this analysis can improve situational awareness to understand the expected behaviour of home IoT devices; providing consumers with a means to interpret what information is communicated beyond their home network, and how their personal data is used. This paper makes the following contributions:

- We propose the use of one-class learning for profiling home IoT devices, and demonstrate this to identify attacks based on anomaly detection.
- We identify networking characteristics of common home IoT devices, and evaluate the ability to define both normal and anomalous networking activity for such devices.
- We present a lightweight IoT profiling tool based on a Raspberry Pi 4, that can be offered to non-technical consumers as a low cost home defence system.

II. Related Works

We position our work in the context of existing research on IoT device profiling, IoT-based machine learning, unsupervised learning anomaly detection, and network-based anomaly detection. Our underlying hypothesis is that IoT network traffic can be distinguished from other types of home network traffic such as laptops and smartphones, as IoT devices will communicate in a more regular and defined manner with a limited number of endpoints. Due to a constrained set of functionality, IoT devices will have more predictable and structured network activity and therefore any changes to this pattern should be noticeable. Meidan *et al.* [17] were able to use a machine learning algorithm to analyse network flows in order to accurately predict which IPs belonged to IoT devices compared to smartphones and PCs. Apthrope *et al.* [18] demonstrate that network traffic rates from IoT devices can leak information regarding user activities.

A lightweight profiling tool aimed at consumer networks must be able to handle high bandwidth traffic, yet still provide real-time protection, therefore resource intensive actions such as deep packet inspection and stateful protocol examination can be avoided by using network flow analysis. Flow-based analysis uses metadata gathered from network communication. Work by Ullah and Mahmoud [19], and Casas *et al.* [20] shows that analysing network flow-based features can be effective for intrusion or activity detection systems. Whilst network flow methods can be faster than packet and stateful protocol examination, common attacks embedded within packets such as SQL injection and XSS would not be detected.

Anthi *et al.* [21] propose a supervised approach to detect a range of popular network based cyber-attacks on IoT networks: Denial of Service (DoS), Man-In-The-Middle (MITM)/Spoofing, Reconnaissance, and Replay. Whilst they do show encouraging results, the nature of supervised learning required significant labelling effort upfront, and could also prove restrictive to identifying new attacks that may arise from new devices or the discovery of new vulnerabilities.

Doshi *et al.* [11] tested five different supervised machine learning algorithms to distinguish normal IoT packets from DoS attack packets. They compared results from an individual packet-based analysis with an approach that analysed two features of a network flow in a rolling 10 second window. Results showed using packet-level machine learning, DoS could be accurately detected and including network flow features further improved accuracy by an F1 score of 0.01 to 0.05.

There are a limited number of IoT-based intrusion detection or botnet datasets available. Some prior works use traditional network datasets such as KDD99 [22], UNSW-NB15 [23] and TUIDS [24] for evaluating their models. However these are not appropriate for analysing IoT networks as the traffic patterns of IoT devices compared with typical computing devices will differ. Koroniotis *et al.* developed a Bot-Iot dataset using simulated IoT sensors [25]. The Avast AIC laboratory created the IoT-23 dataset [26] containing 20 malware captures from various IoT devices. Both datasets only provide raw PCAP data that represents malicious test cases, rather than separate PCAP data for both malicious and benign activity. Our proposed model requires a suitable period of only benign data for it to be formative of deviations caused by malicious variations. Furthermore, the datasets alternatively provide a labelled text-based version of the same activity, however this only provides summary detail where they omit many of the key features that are utilised in our model.

In our research, Legg [27] studied the effectiveness of visual analytics for engaging non-technical users in home networking security, recognising that traditional security tools do not effectively cater for non-technical users in a way that encourages security culture and adoption. Our research on IoT device profiling highlights that much prior work addresses supervised machine learning methods. Here, we specifically address unsupervised one-class learning due to the prevalence of class imbalance in security monitoring, as well as to reduce the upfront burden of class labelling, and also providing flexibility to address unknown attack vectors in the future.

III. Experimentation

For the purpose of this study, we examine a typical home IoT network. Specifically, we omit traditional computing devices and smart phones, and focus on five unique IoT devices: Amazon Echo, Amazon Fire TV, Brother Printer, Netgear Arlo Security Camera, and Home Hive Hub. The Amazon devices and the printer are connected via Wi-Fi, and the camera system and Hive Hub are connected via Ethernet. These devices were used within the home environment of the first author; therefore, the normal behaviour reflects the behaviour of the first author. Future research would explore the feasibility of developing aggregated benign models in a lab environment.

A network monitoring device was developed using a 4GB Raspberry Pi 4. Figure 1 displays an overview of the testbed. A Netgear GS108T switch was used to port mirror all network traffic from the home router to the Raspberry Pi Ethernet

adaptor, which was placed into the promiscuous monitoring mode. The Raspberry Pi was also connected via Wi-Fi so that the web application could be accessed for monitoring and management purposes.

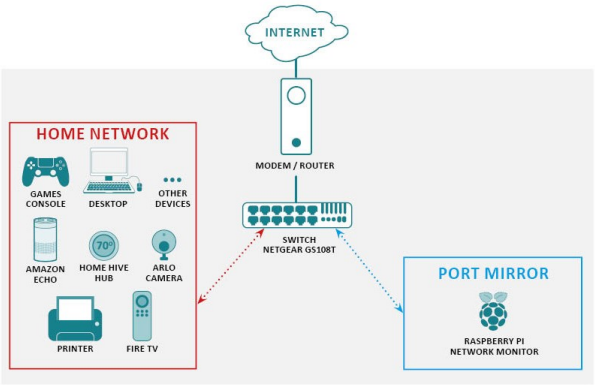

Fig. 1. IoT Testbed configuration

A. Network Monitor

The network monitoring device periodically scanned the local network to ascertain which devices were on the network and online. Simple device identification was performed based on MAC vendor and service fingerprinting to detect likely IoT devices. The IoT device IPs were fed into the packet capture module in order to limit the amount of traffic being captured to that of the IoT devices on the network. As the packet capture processing is performed on a resource-constrained device, reducing the load on the Raspberry Pi is key in maintaining real-time performance.

B. Packet Capture

The packet capture module uses *tcpdump* to collect network traffic from the port mirror data received at the Ethernet interface. Packet captures files are rotated and processed every 60 seconds. The specific features extracted and analysed from the packet capture are detailed in section III-D. For each device, the extracted data is processed and stored in a database.

When a device is initially discovered, an initial two-day period is used to obtain sufficient training data where device behaviour is monitored and profiled. This makes a clear assumption that the device is initially operating normally, rather than being compromised from the outset. The captured data is saved, and used to fit a machine learning model. After this training period, new traffic captures are compared to the model to check for anomalous behaviour every 60 seconds.

Information is visualised for the user on a node-information web page. Data is aggregated into 15-minute periods for display purposes. Total Data In/Out is shown, along with graphs depicting the average packet size, average number of packets per conversation and average data per conversation. It is hypothesised that these graphs will show a difference in behaviour during normal operation versus during an attack. The top 5 inbound and outbound connections based on data transferred are displayed, showing the IP address involved, the name of the company that this resolves to via a WHOIS lookup, and the amount of data transferred. It would be expected that an Amazon device would be mostly conversing with Amazon servers. A change in this behaviour could be an indicator of an attack of some kind taking place. Figure 2 shows an example of these graphs for an Amazon Echo device exhibiting normal user behaviour.

Fig. 2. Normal activity behaviour for an Amazon Echo

C. One-Class Support Vector Machine

Based on the traditional support vector machine, the One-Class Support Vector Machine (OC-SVM) algorithm was developed to resolve the training classification problem which only has one type of sample. The model can be trained with benign traffic to build a picture of the normal communication behaviour of a device and used as a base for anomaly detection. OC-SVM maps the input space into a high-dimensional space by the kernel function. Popular choices for the kernel function are linear, polynomial, sigmoidal and radial base function. The Support Vector Method For Novelty Detection by Schölkopf *et al.* [28] separates all the data points from the origin and maximises the distance from the hyperplane to the origin. This results in a binary function that captures regions in the input space where the value +1 is a small region capturing most of the data points, and -1 elsewhere. In Figure 3 we can see an example of the learned frontiers of a set of two dimensional data points.

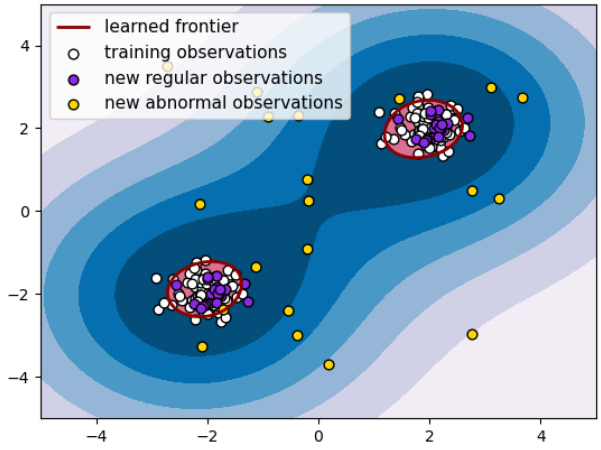

Fig. 3. Example OC-SVM plot illustrating learned frontiers [29]

Feature Name	Description
Protocol	Protocol Type (Eg, UDP, TCP, ICMP, etc)
Address A	Address of the monitored target
Address B	Remote host IP Address
Port A	Monitored target port number
Port B	Remote host port number
Packets A → B	The number of packets from monitored target to remote host during the conversation
Bytes A → B	The number of bytes from monitored target to remote host during the conversation
Packets B → A	The number of packets from the remote host to the monitored target during the conversation
Bytes B → A	The number of bytes from the remote host to the monitored target during the conversation

TABLE I
METHODS USED FOR CURATING NETWORK TRAFFIC ANALYSIS FEATURES

D. Data Collection

This section presents the data collection, feature extraction and machine learning classifier phase of the research. We use the OC-SVM from the popular scikit-learn library [29].

Several basic, traffic and connection-based features were extracted from each packet. The data is aggregated from each network capture file into unique data flows between a source and destination based on the protocol, source and destination IP addresses and the source and destination port values. We refer to this grouping of data communication as a conversation. As network traffic is often encrypted, the payload information from the Application Layer in the message is not considered as a feature. Table I shows the list of extracted features. When the data is aggregated, the Address A value is set as the monitored target on the local network.

Once the training period is complete, the pre-processed data was analysed to remove any outliers and erroneous records. The IP addresses for Address A and Address B were converted to a decimal value and the values of all the features were standardised by removing the mean, and scaling to unit variance using the scikit-learn StandardScaler. This was required in order to remove the variance in features such as the IP addresses used. The scaled data is then passed though the scikit-learn Principal Component Analysis (PCA) to perform linear dimensionality reduction, based on 3 component features of the data.

Once the training period is complete and a model is created, the packet capture is rotated and processed every 60 seconds. Current device data is assessed against the model to identify potential outliers, which are then reported to the user if a threat is found. As the model is trained with only benign network traffic, the positive class is defined as benign traffic, and the negative class defined as the malicious traffic.

IV. Attacks

Recent studies [9], [3] show that IoT devices have been compromised as botnets to form distributed denial-of-service attack networks. Therefore, we focus this work on the detection of DoS traffic patterns. Specifically for our testing, two forms of DoS traffic will be generated using *hping3* on a Kali Linux machine that is connected to the same network as the IoT devices under test, which are:

- **TCP SYN Flood:** An attacker sends SYN messages to the target from spoofed source IP addresses. The SYN, ACK response returned by the server is sent to a fictitious host. This causes the target to use their resources waiting for the ACK from the originator. The number of TCP connections is limited on the target, so, if enough SYN packets are sent, it prevents the target from accepting new connections.
- **UDP Flood:** UDP floods send a large amount of data from spoofed IP addresses to the target. If enough data is sent, it will consume all available bandwidth on the server.

A common duration for DDoS attacks is around 90 seconds, so as to try and avoid detection [11]. Therefore our attack simulation will run for a maximum of 2 minutes at a time.

V. Evaluation Metrics

Several measures can be used to evaluate the performance of a machine learning classifier:

- **True Positives (TP):** Conversations predicted as being benign that are indeed benign.
- **True Negatives (TN):** Conversations predicted as malicious that are indeed malicious.
- **False Positives (FP):** Conversations predicted as benign that are actually malicious.
- **False Negatives (FN):** Conversations predicted as malicious that are actually benign.

When we use a one-class classification, we train the model with only "positive" class data, that being benign network data. Since our training data is all benign, the output of the model could only result in TP or FN cases, since we have no malicious data to inform TN or FP cases.

When assessing the DoS attack traffic against the model we are looking for True Negatives and False Positives. True Negatives should be the DoS traffic correctly identified as an outlier and malicious traffic. False Positive is attack traffic that has been identified as benign. Although the attack data

is unlabelled, the data is known by us to be negative class by the fact that we know the PCAP provided contains only DoS traffic. As we are focused on detection of the negative class, the evaluation metrics detailed in Table II can be used to assess the performance.

Accuracy can be used as a measure of performance. Accuracy measures the number of conversations that were correctly classified. However, accuracy does not tell the full story when working with a class imbalanced data set. If 90% of the traffic is benign, the classifier can achieve a 90% accuracy rate by always classifying the traffic as benign, whereas in reality, it failed to detect 100% of the malicious cases.

Precision attempts to answer the question *"What proportion of negative identifications was actually correct?"* whilst recall attempts to answer *"What proportion of actual negatives was identified correctly?"*. To fully evaluate the effectiveness of a machine learning model, both precision and recall methods must be analysed. These measures are used together in the F-measure, which provides a single weighted metric to evaluate the overall classification performance. An F-measure value of 1 is perfect precision and its worst value is 0.

VI. Results and Discussion

A. Machine Learning Classifier

For each device, the dataset is collated for a training period of two days. The training data was fitted to create an OC-SVM model of normal network traffic behaviour for this user/device. A unique model was created for each IoT device under test and stored on the Raspberry Pi. During testing, the Amazon Fire TV device caused issues with the stability of the system due to the large volumes of data being collected. All the other IoT devices under test generate low bandwith traffic. The Amazon Fire TV was streaming UHD video streams for long periods, generating gigabytes worth of traffic. The Raspberry Pi and Python scripts were unable to cope with the high volume of traffic and process it in a reasonable time frame, and so the device was removed from further testing.

Figure 4 shows the distribution of benign data flow conversations used for training the model against the number of DoS attack conversations collected that were used to predict against the model. This demonstrates the uneven balance of classification of the packets. The Home Hive and Arlo devices show less imbalance in benign versus attack conversations due to the low amount of traffic that these devices generate during normal operation.

Due to the issues previously discussed regarding network packet capture for large data streams, the DoS attacks were throttled from line rate to a level where the Raspberry Pi could process the packets within the packet rotation period of 60 seconds. A line-rate attack would cause a capture file of approximately 30MB to be generated in one minute, causing the monitor to fail. The value of 100 packets a second was the highest value that the monitor was able to capture and process in addition to the regular network traffic for the device.

The hyper-plane variables were adjusted and different kernels of 'linear', 'rbf', 'poly' and 'sigmoid', with different

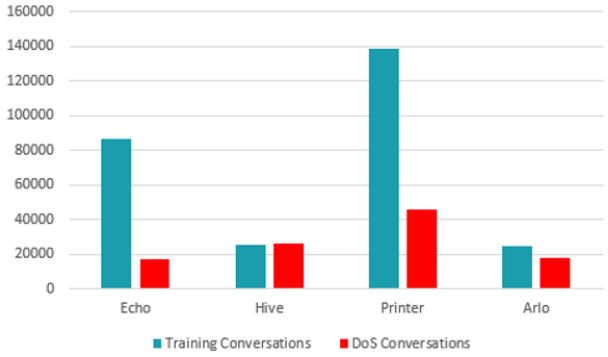

Fig. 4. Distribution of packets

gamma and nu values were tested until satisfactory results were achieved. The final values selected for the model were:

- **nu** = 0.000005 (An upper bound on the fraction of training errors and a lower bound of the fraction of support vectors.)
- **kernel** = rbf (Radial basis function)
- **gamma** = 0.000000001 (kernal coefficient)

Table III reports the overall weighted-average performance for the classifier against the devices under test, using the evaluation metrics previously described in Table II, resulting in F-measure scores ranging between 91.58 to 99.67. To gain a better insight into the performance of the classifier for each device, Table IV shows the confusion matrix for how the predicted classes for conversations compares against the ones analysed for the model fit and attack detection. Figure 5 shows the plot of the OC-SVM clustering plotted with the two dimensions with the highest explained variance ratio after the PCA dimensionality reduction. This shows a clear difference in clustering for benign and malicious traffic patterns for all four devices under test.

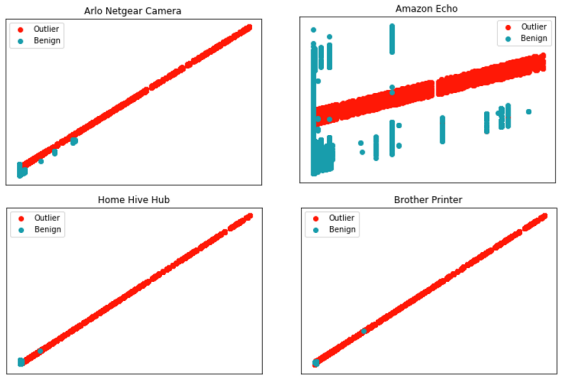

Fig. 5. Plot of the OC-SVM clustering results.

B. Amazon Echo

Normal behaviour data flow for the Amazon Echo shows that there is minimal data transfer during idle periods, and

Metric	Calculation	Value
Accuracy (A)	$\frac{TP+TN}{TP+TN+FP+FN}$	Proportion of predictions the model got right
Precision (P)	$\frac{TN}{TN+FN}$	Proportion of conversations identified as malicious that was correct
Recall (R)	$\frac{TN}{TN+FP}$	Proportion of malicious conversations identified correctly
F-Measure (F)	$F = 2 \cdot \frac{P \cdot R}{P+R}$	Harmonic mean of Precision and Recall

TABLE II
EVALUATION METRICS

Device	Training Size	DoS Size	Accuracy	Precision	Recall	F-Measure
Amazon Echo	86549	17080	96.99	85.00	99.27	91.58
Home Hive Hub	25225	26159	98.21	98.87	98.27	98.57
Brother Printer	138682	45954	96.46	99.38	99.77	99.67
Netgear Arlo	24329	18077	96.23	93.33	98.17	95.69

TABLE III
WEIGHTED AVERAGE FOR EACH IOT DEVICES

			Predicted	
	IoT Device		**Benign**	**Malicious**
Actual	Amazon Echo	**Benign**	86549	5920
		Malicious	140	16940
	Home Hive Hub	**Benign**	25225	9
		Malicious	9315	16844
	Brother Printer	**Benign**	138682	1075
		Malicious	1529	44425
	Netgear Arlo	**Benign**	24329	3077
		Malicious	6010	12067

TABLE IV
MODEL FIT AND ATTACK DETECTION CONFUSION MATRIX

limited outgoing traffic from the device, only voice activated queries and packet acknowledgements to the incoming streamed data. The average packet size and data per conversation show a pattern for normal usage, and the graphs reflect a minimal change in behaviour during a DoS attack (Figure 6). Average packet size drops compared to normal idle usage, but there is no significant difference between the average number of packets in a conversation, nor the average data per conversation which is reflected in the OC-SVM clustering graphs in Figure 5.

The nominal source IP addresses for typical user behaviour were Amazon AWS servers and Content Delivery Networks (CDNs) such as Limelight and Akamai. During streaming audio operation, the service changes the origination IP address of the data after approximately 100KB of data. During streaming audio for multiple hours, over 1000 unique IP addresses associated with the content delivery service were used. This deviated from the expected behaviour as the original hypothesis for content delivery would be that an entire stream would be served from a single IP address and that DoS traffic would stand out as abnormal by originating from lots of IPs.

During a DoS attack, the attack originates from thousands of spoofed IP addresses, so, although the number of IP addresses associated with communication has increased, the constant change of IP address, coupled with the similar attributes of conversations during idle periods, make the attacks difficult to see via the graphs alone. The classifier returned an F-measure score of 91.58%, showing that it was still able to determine the DoS traffic compared to normal behaviour with high accuracy.

Fig. 6. Amazon Echo data usage with DoS attack data highlighted

C. Home Hive and Water Sensor

The connected water leak sensor was hardly ever discovered online during a host scanning. The device operates in an ultra-lower power mode and only powers up the Wi-Fi interface when a leak has been detected. It was only detected once online during a week of network scans. The hub remained online throughout the test and had a constant low-level flow of traffic, averaging about 300 bytes per 15-minute aggregated period between itself and servers hosted by Amazon.

The consistently low level of normal traffic enables any kind of network-based attack to stand out as unusual. As anticipated, the Data In volume of traffic has increased significantly compared to the normal level of data, and the average packet size has decreased. The average number of packets and data per-conversation has trended towards zero compared to normal

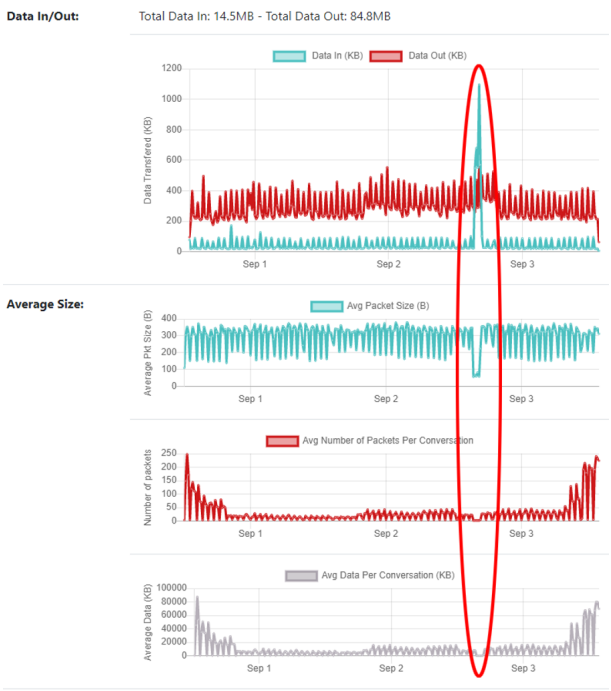

Fig. 7. Home Hive Hub normal data usage with DoS attack data highlighted

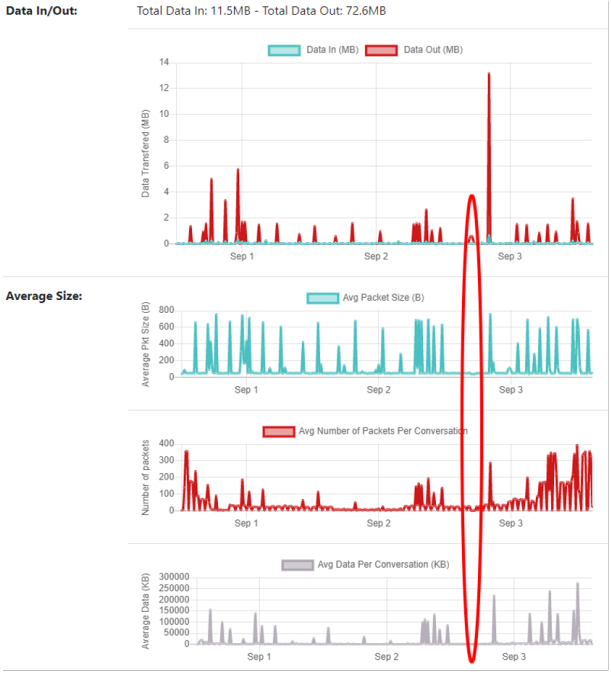

Fig. 8. Arlo Netgear Camera normal data usage with DoS attack data highlighted

behaviour, due to the single 60-byte packets being sent from each IP address in the attack. The results from the classifier shows an F-measure score of 98.57%.

D. Netgear Arlo Security Camera

The Netgear Arlo cameras send 10-second video files to an Amazon cloud server each time motion is detected. Netgear content servers are hosted by Amazon AWS, so all legitimate communication takes place to these servers. Inbound data to the device is negligible, but outbound data is punctuated by 1.7MB files being sent to the servers each time motion is detected. Each file is sent to a different Amazon AWS IP.

As the DoS attack rate was rate-limited due to monitoring limitations, the graph scales do not reflect the attack. A marginal drop in the average size of packets and communication can be seen, but only if the user knows to look for this. The classifier returned a F-measure score of 96.69%.

E. Brother Printer

Normal network traffic, both inbound and outbound from the device, is internal network traffic only. Only devices configured with UPnP, or devices that send print jobs to the printer are involved in normal traffic flows. The amount of data transferred within a 15-minute monitoring period is in the order of $<$100 Kilobytes. Any DoS attack stands out as abnormal on the graph. The classifier returned an F-measure of 99.67%, demonstrating it can successfully distinguish between malicious or benign traffic with high-accuracy.

VII. Conclusions

A. Challenges in Network Monitoring

The most important design considerations for creation of such a monitor are associated with performance. The biggest limitation in this research is the performance of the Raspberry Pi in processing the packet captures. We used a Pi to address how non-technical users could adopt a low-cost solution for monitoring home IoT networks. However the inability of the Scapy Python package to be able to process packet captures of greater than 2.5MB within 60 seconds renders the device incapable of being able to be deployed as a general real-time network monitor.

When the traffic monitoring was limited to low-bandwidth IoT devices, using an unsupervised OC-SVM classifier to analyse network traffic for anomalies shows that it is highly effective in detecting the DDoS threats tested. The performance of the unsupervised machine learning resulted in an F-measure across devices of between 91.58% to 99.67%. This demonstrates that the proposed architecture can successfully distinguish between abnormal traffic from a normal traffic profile that was learnt during an unsupervised training period. Packet length and number of packets in a conversation were good indicators of malicious behaviour; specifically when the packets are significantly smaller than normal.

A limitation to the approach of building a normal profile of device behaviour from an unsupervised training period is that we have to assume that the device is not compromised before it is added to the network. If a compromised device is added to the network, the monitor will learn the malicious behaviour as normal behaviour. A solution to this would be to pre-define

models for popular IoT devices. However, this would require on-going support to define and maintain these profiles in line with device software updates.

B. Recommendations and Future Work

The research showed that IoT communication to CDNs differed in behaviour from what was initially expected. It was expected that a data stream would be from a single IP address, and therefore DoS attack would stand out as arriving from a large number of destinations. However, it was shown that an Amazon Echo music stream would download ˜90KB from a single CDN IP before changing IP address, and would use 1000s of unique IPs for a long stream. Instead of using the source IP address as a feature, the ASN number associated to the IP could be used. This would provide less variability for normal behaviour, as the ASNs would be associated to Amazon or CDNs, but show a marked difference in behaviour for malicious behaviour such as a DDoS. Further attack types should be tested, and a second layer of machine learning could also be added once malicious behaviour has been detected, in order to classify the type of attack that has been received. The cluster plots from the OC-SVM model show that the DoS traffic has a clear pattern of attack in all four devices. It is anticipated that other attacks will show different clustering and therefore the attack types can be distinguished. This research has not yet explored how the analysis of attack data is best presented to the user. To what extent do we want to present transparency of the ML decisions to the home user? The output of classifier does not currently present why the classifier has determined the traffic to be abnormal. How does a home user understand the ML-based outcome, and how do we inspect the network behavioural change? At the moment the data is presented in a high-level view to the user by means such as data in/out graphs, but future research should investigate how to engage and educate non-technical users on IoT device security.

REFERENCES

[1] M. Fagan, K. N. Megas, K. Scarfone, and M. Smith, *Foundational cybersecurity activities for IOT device manufacturers*. US Department of Commerce, National Institute of Standards and Technology, 2020.

[2] Owasp-iot-top-10-2018. Last accessed 27 April 2021. [Online]. Available: https://owasp.org/www-pdf-archive/OWASP-IoT-Top-10-2018-final.pdf

[3] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "Ddos in the iot: Mirai and other botnets," *Computer*, vol. 50, pp. 80–84, 2017.

[4] Smart home - worldwide, statista market forecast. Last accessed 27 April 2021. [Online]. Available: https://www.statista.com/outlook/283/100/smart-home/worldwide

[5] O. Alrawi, C. Lever, M. Antonakakis, and F. Monrose, "Sok: Security evaluation of home-based iot deployments," in *IEEE Symposium on Security and Privacy*, 2019, pp. 1362–1380.

[6] S. Notra, M. Siddiqi, H. Gharakheili, V. Sivaraman, and R. Boreli, "An experimental study of security and privacy risks with emerging household appliances," in *IEEE Symposium on Security and Privacy*, 2014, pp. 79–84.

[7] T. Yu, V. Sekar, S. Seshan, Y. Agarwal, and C. Xu, "Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the internet-of-things," in *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*, 2015, pp. 1–7.

[8] S. Raza, L. Wallgren, and T. Voigt, "Svelte: Real-time intrusion detection in the internet of things," *Ad hoc networks*, vol. 11, no. 8, pp. 2661–2674, 2013.

[9] E. Bertino and N. Islam, "Botnets and internet of things security," *Computer*, vol. 50, no. 2, pp. 76–79, 2017.

[10] M. Yamauchi, Y. Ohsita, M. Murata, K. Ueda, and Y. Kato, "Anomaly detection in smart home operation from user behaviors and home conditions," *IEEE Transactions on Consumer Electronics*, vol. 66, no. 2, pp. 183–192, 2020.

[11] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning ddos detection for consumer internet of things devices," in *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2018, pp. 29–35.

[12] M. V. Mahoney, "Network traffic anomaly detection based on packet bytes," in *Proceedings of the 2003 ACM symposium on Applied computing*, 2003, pp. 346–350.

[13] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection," in *Proceedings of the 2003 SIAM international conference on data mining*. SIAM, 2003, pp. 25–36.

[14] P. Sharma, K. Chaudhary, M. Wagner, and M. G. Khan, "A comparative analysis of malware anomaly detection," in *Advances in Computer, Communication and Computational Sciences*. Springer, 2020, pp. 35–44.

[15] A. Mills, T. Spyridopoulos, and P. Legg, "Efficient and interpretable real-time malware detection using random-forest," in *2019 International Conference on Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA)*, 2019, pp. 1–8.

[16] P. A. Legg, O. Buckley, M. Goldsmith, and S. Creese, "Automated insider threat detection system using user and role-based profile assessment," *IEEE Systems Journal*, vol. 11, no. 2, pp. 503–512, 2017.

[17] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "Profiliot: a machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the symposium on applied computing*, 2017, pp. 506–509.

[18] N. Apthorpe, D. Reisman, and N. Feamster, "A smart home is no castle: Privacy vulnerabilities of encrypted iot traffic," *arXiv preprint arXiv:1705.06805*, 2017.

[19] I. Ullah and Q. H. Mahmoud, "A two-level flow-based anomalous activity detection system for iot networks," *Electronics*, vol. 9, no. 3, p. 530, 2020.

[20] P. Casas, J. Mazel, and P. Owezarski, "Unsupervised network intrusion detection systems: Detecting the unknown without knowledge," *Computer Communications*, vol. 35, no. 7, pp. 772–783, 2012.

[21] E. Anthi, L. Williams, M. Słowińska, G. Theodorakopoulos, and P. Burnap, "A supervised intrusion detection system for smart home iot devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9042–9053, 2019.

[22] Kdd cup 99 dataset. Last accessed 27 April 2021. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[23] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[24] P. Gogoi, M. H. Bhuyan, D. Bhattacharyya, and J. K. Kalita, "Packet and flow based network intrusion dataset," in *International Conference on Contemporary Computing*. Springer, 2012, pp. 322–334.

[25] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[26] A labeled dataset with malicious and benign iot network traffic. Last accessed 27 April 2021. [Online]. Available: https://www.stratosphereips.org/datasets-iot23

[27] P. A. Legg, "Enhancing cyber situation awareness for non-expert users using visual analytics," in *2016 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (CyberSA)*, 2016, pp. 1–8.

[28] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, J. C. Platt *et al.*, "Support vector method for novelty detection." in *NIPS*, vol. 12. Citeseer, 1999, pp. 582–588.

[29] sklearn.svm.oneclasssvm - scikit-learn 0.24.2 documentation. Last accessed 27 April 2021. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html