Article

Network Anomaly Detection inside Consumer Networks—A Hybrid Approach

Darsh Patel [1], Kathiravan Srinivasan [2], Chuan-Yu Chang [3,*], Takshi Gupta [4] and Aman Kataria [5]

[1] School of Computing Science and Engineering, Vellore Institute of Technology, Vellore 632014, India; darshkpatel@gmail.com
[2] School of Information Technology and Engineering, Vellore Institute of Technology, Vellore 632014, India; kathiravan.srinivasan@vit.ac.in
[3] Department of Computer Science and Information Engineering, National Yunlin University of Science and Technology, Yunlin 64002, Taiwan
[4] Information Security Engineering, Soonchunhyang University, Asan-si 31538, Korea; takshi_gupta2012@hotmail.com
[5] Department of Electrical and Instrumentation Engineering, Thapar Institute of Engineering and Technology, Patiala 147004, India; ammankataria@gmail.com
* Correspondence: chuanyu@yuntech.edu.tw

check for
updates

Abstract: With an increasing number of Internet of Things (IoT) devices in the digital world, the attack surface for consumer networks has been increasing exponentially. Most of the compromised devices are used as zombies for attacks such as Distributed Denial of Services (DDoS). Consumer networks, unlike most commercial networks, lack the infrastructure such as managed switches and firewalls to easily monitor and block undesired network traffic. To counter such a problem with limited resources, this article proposes a hybrid anomaly detection approach that detects irregularities in the network traffic implicating compromised devices by using only elementary network information like Packet Size, Source, and Destination Ports, Time between subsequent packets, Transmission Control Protocol (TCP) Flags, etc. Essential features can be extracted from the available data, which can further be used to detect zero-day attacks. The paper also provides the taxonomy of various approaches to classify anomalies and description on capturing network packets inside consumer networks.

1. Introduction

Multiple sweeping attacks on key Internet services around the world have been launched with botnets powered by Zombie Internet of Things (IoT) devices such as security cameras and wireless routers, with attack bandwidth topping 1.1 Terabits per second [1]. This shows that compromised IoT devices can pose a huge threat if hacked successfully. Not only that, but they also act as a pivot point inside a network from where perpetrators can exploit other devices, sniff sensitive information from the local network, and use device resources to mine cryptocurrencies, etc.

Today's commercial intrusion detection systems are primarily signature-based, which means they depend on predefined signatures of known attacks or carefully set up rules to filter out any possibility of attacks [2]. These require frequent signature updates and operators proactively update rules for these systems to work effectively. Thus, without frequent updates, these devices fail to detect the latest threats, not to mention they cannot protect against Zero-Day attacks due to their inherent nature of having no previous instances of them being used and hence no signatures to compare [3].

Network Anomaly detection is a wide topic which has been studied in numerous research, articles, surveys as well as books [4–6]. A major part of existing works aims at thwarting attacks on commercial networks [7]. One of the most common approaches to detecting anomalies is Outlier detection which has been studied by the statistical community [4,8–16], but with recent advancements in machine learning, it has since been playing a notable role in anomaly detection. While it seems attractive conceptually, this approach has its own set of drawbacks, such as high false-positive rates to the intrinsic complexity of the system, determining which exact event triggered the alarm, etc. These problems need to be addressed before a wide adaptation of anomaly-based detection systems.

To help understand the advantages of an intelligent system in anomaly detection, this paper discusses a hybrid approach based on the One-Class Support Vector machine (OCSVM) using the Entropy of various network features to classify anomalies. It also provides a definite approach for collecting and analyzing network traffic, also portraying essential features that can be extracted from this data; and specifically used for marking zero-day vulnerabilities. The performance and accuracy of the models, One-Class Support Vector Machine, and Isolation Forest have been compared in this paper. The models were trained on the features derived from Table 1 and have been listed in Table 2. The results have been mentioned in Tables 3 and 4 and have been discussed in Section 4.

1.1. Motivation and Problem Statement

The number of IoT devices connected to the Internet is projected to grow to 28 billion by 2021 [17]. Most of these IoT devices are inherently insecure due to common default passwords, open and unauthenticated telnet ports, outdated and unpatched Linux firmware, and unencrypted transmission of sensitive data, etc [18]. With the increase in the number of Internet-connected devices, the number of attacks targeting such devices has increased at the same time.

Identification of common threats because of intelligent embedded codes is tedious, and it is required to taken certain steps to increase the identification of threats by leveraging the properties of machine learning and intelligent solutions [19,20]. These attacks become more dominant when targeted inside consumer households. These households do not consist of dedicated firewalls or managed switches with monitoring capabilities; thus, novel solutions that can run on low powered devices need to be developed which allow detection and prevention of compromised devices or attacks.

1.2. Distinction from Existing Research

Network anomaly detection research has been targeted towards commercial scenarios. Commercial networks are a target for a multitude of attacks that are not a concern for a consumer network.

Distributed Denial of Service (DDoS) attacks are generally aimed at essential internet infrastructure and websites, and mitigating such attacks has been a challenge taken up by many researchers. The key reason being the fact that a large portion of the attack bandwidth comes from "Zombie" devices found in consumer networks. There is research to detect DoS attacks on the LAN inside consumer networks [21], but very little light has been shed on detecting if a device on the network is acting as a part of a botnet to execute large scale DDoS attacks.

Generally, the comparison is made using the DARPA's KDD'99 Dataset [11,14,22]; however, this dataset is not only outdated but cannot be used to evaluate machine learning algorithms for network anomaly detection inside consumer networks. The dataset containing traffic from eight households (Section 3.4.3) and also synthetic attacks were utilized trying to simulate the real-world scenario as close as possible. Details about capturing and processing data are discussed in Section 3.4.2.

Detecting anomalies on a consumer network has its own set of challenges:

- Lack of data points: Consumer networks do not have subnets containing hundreds of devices generating a multitude of traffic over various protocols which can be analyzed for patterns. For example: In a typical work environment, one can monitor traffic on the DC analyzing user

login data or the average overall bandwidth use whose patterns correlate work hours; which cannot be done on a typical consumer network.

- Unavailability of public datasets: For commercial networks, publicly available labeled datasets with millions of records and attack types have been made available for research purposes [23]. There has also been work done on the quality of such datasets. However, consumer network data have not been collected or published for privacy reasons.

- Lack of infrastructure: There aren't managed switches or routers with monitoring capabilities in a consumer network. In such networks, packet capturing tools need to be installed on users' computers or a low power device like a Raspberry Pi has to be installed on the network to monitor the network.

2. Anomaly Detection Methods

Details of different types of anomaly detection methods are provided below:

2.1. Supervised Classification Based Anomaly Detection

This approach is a supervised learning approach where the model is trained using a labeled dataset, which then tries to classify new data based on the training data. Linear classification tries to find a line between the classes [24], but the classification boundary may be nonlinear too [24], as seen in Figure 1. These techniques have a low false-positive ratio subject to suitable thresholds [24]. The prime drawback of these is that they are highly dependent and biased on the training data and thus generally cannot identify anomalies it has not observed before, which defeats the purpose of this approach.

2.2. Statistical Anomaly Detection

Any irrelevant entity which is not generated by the stochastic model is considered as an anomaly in this model as stated in [24]. Thus, the occurrences with a low probability of being generated are treated as anomalies. The prime advantage of this technique over the others is that it does not require "Prior Knowledge" of the network's normal activity [6]; thus, it can provide accurate results about anomalous activity [24]. One promising approach to detect network anomalies is an Entropy-based approach. Entropy is a measure of the uncertainty or randomness associated with a random variable. If it was more random, it contains more entropy [25]. The primary drawback of this approach is that attackers can "train" the detection model until the traffic is considered as normal according to the statistical model. This approach also requires a lot of training data to create baseline entropy for each feature being targeted, which is difficult to obtain in a consumer setting. Since the margins for triggering alerts based on entropy values have to be manually set, there is room for error, and optimization of these values is difficult.

2.3. Clustering and Outlier Based Novelty Detection

Clustering and outlier approaches can be studied as unsupervised and supervised approaches as explained below:

2.3.1. Unsupervised Approach

Fundamentally, grouping data into various sets of similar objects is called clustering, as represented in Figure 2a. In anomaly detection, the primary assumption made is that the larger clusters are normal and the rest of the clusters can be considered as anomalies [24]. In Figure 2b, the points which do not fit into any of the clusters are considered outliers (anomalous data–points). It is also worth noting that the unsupervised learning approach works with unlabeled data. Due to most unsupervised learning models using both outlier detection and clustering, the computation complexity can be quite high [24]. The prime advantage over other techniques is that the outliers can be detected with small datasets. In addition, due to its nature, small isolated bursts can be detected.

Being closely related to statistical models, this approach also suffers from attackers being able to "train" the detection model until the traffic is classified as normal.
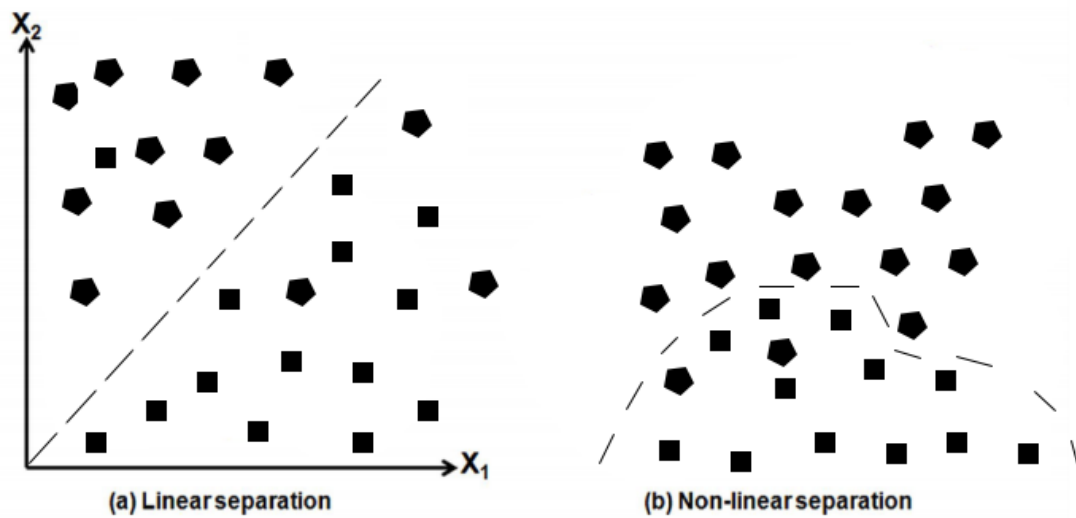

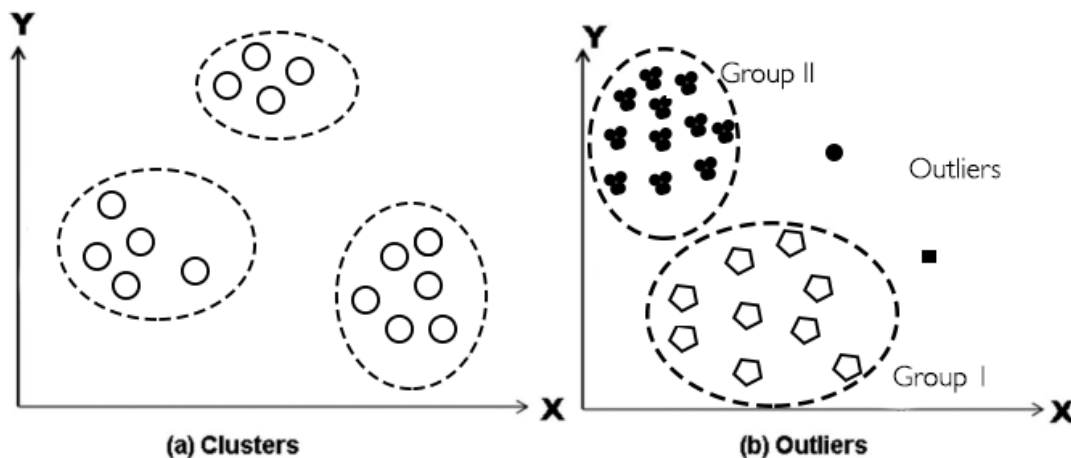**Figure 1.** Classification based anomaly detection from [24].

Figure 2. Clustering and outlier based anomaly detection [24].

2.3.2. Semi Supervised Approach

Semi-supervised learning is prevalent in scenarios where there is very little anomalous data available to train the model, but non-anomalous data are readily available, thus it is trained on a single class of data and then detects novelties. Thus, this is also commonly referred to as novelty detection [26]. One-Class Support Vector Machine classifiers (OCSVM) are favorable in case of anomaly detection as they do not require pre-labeled data sets which are expensive or difficult to obtain [27].

3. Proposed Hybrid Approach for Anomaly Detection

Since network usage patterns might keep changing inside a household, e.g., kids browsing a lot of different sites on holidays or while doing homework, relying only on the entropy of network traffic features can be unreliable. This paper proposes to use the normalized entropy of features mentioned in Table 2 to be processed by One-Class SVM for anomaly detection inside consumer networks.

3.1. Application Model

A modern household can be considered as an application scenario equipped with smart televisions, smart refrigerators, home automation systems, and other connected embedded devices. These devices, when compromised, can be utilized as zombies for DDoS Attacks [28]. In the setup used in this paper, the network data were captured from eight households with varying numbers and types of IoT devices. Details about test setup are mentioned in Section 3.4.2.

3.2. Entropy

Entropy is the measurement of uncertainty. One can say that the network entropy tends towards 1 when there is a lot of randomness in the traffic. For example, in a Distributed Denial of Service (DDoS) attack period, the entropy of source Internet Protocol (IP) will increase due to the number of source IPs of attackers being uncertain [25]. Similarly, destination IP entropy decreases as most packets that will have the destination IP will be those of the victim. Thus, the entropy values would deviate from its normal baseline values, and that can be classified using the OCSVM model for classification

Using entropy of features for classification with OCSVM can help eliminate most of the demerits of entropy-based anomaly detection as discussed in the explanation of anomaly detection approaches while also utilizing the merits of Support Vector Machine (SVM) since entropy can represent network traffic changes better and One-Class SVM can classify values better.

3.3. Normalized Entropy

By the definition of the entropy [25], it is expressed using the probable values for a given variable (feature in the proposed approach) and its probability distribution. Considering that for a random variable, X, its values are observed for a fixed time window, its probability distribution can be expressed as n_i/n, where n_i is the number of times the value is observed for the variable (feature) and n is the total number of values for the variable (feature). The `Normalized Entropy` is calculated as $H/\log n_0$, where n_0 is the number of distinct values in time interval. The steps for entropy calculation can be followed in Algorithm 1 [25].

Algorithm 1 Normalized entropy

Input: Network traffic for a feature from Table 1
Output: Entropy value for network feature
  for each time interval **do**

    for each packet in time interval **do**

      Calculate Frequency of each distinct value of feature
    end for
    for each distinct value of feature **do**

      $P_i = n_i/N$ n_i = frequency of ith value and N = number of values in time interval
      The normalized entropy $H = \sum h_i/\log(n_0)$
      Where $h_i = -\sum p_i \log p_i$
    end for
  end for

3.4. Methodology

The complete process from packet being captured to classification has been represented in Figure 3; below are the steps followed:

- Capturing Network Packets (Sections 3.4.1 and 3.4.2).

- Calculating entropy for features as expressed in Algorithm 1.
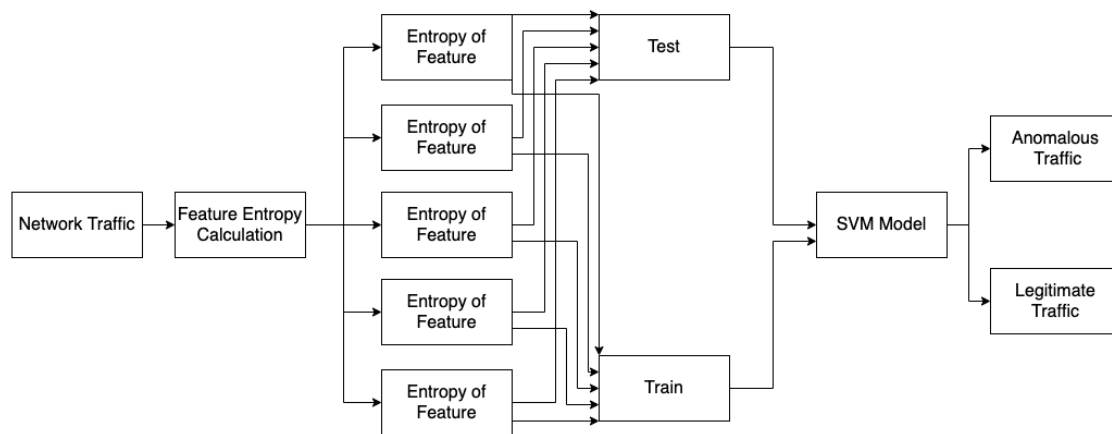- Entropy values are used in OCSVM Model for Classification.

Figure 3. Overview of the proposed approach

Table 1. Data–points collected.

Sl.	Feature Name	Feature Description
1	Protocol	Layer 3 Protocol: IP,User Datagram Protocol(UDP),TCP
2	Source IP	Packet Source IP Address
3	Destination IP	Packet Destination IP Address
4	Source Port	Packet Source Port
5	Destination Port	Packet Destination Port
6	Frame Length	Length of the captured frame
7	Time	Time of said packet
8	Time To Live (TTL)	Packet Time to Live
9	TCP Flags	Flags: Reset (RST),Synchronization (SYN),Acknowledgement (ACK),Finish (FIN)

Table 2. Features extracted from Table 1.

Feature	Feature Description
1	Ratio Between TTL and Payload Size
2	TTL in N seconds
3	Payload Size in N seconds
4	Number of Domain Name System (DNS) queries in T Seconds
5	Number of ACK packets in T seconds
6	Number of RST packets in T seconds
7	Number of FIN packets in T seconds
8	Number of SYN packets in T seconds
9	Number of IPv4 Frames in T Seconds
10	Time between two frames over T seconds

3.4.1. Tools Used for Evaluations

Network sniffer `wireshark` [29] was used to capture network data which was then stored in a PCAP file. The PCAP file which consists a lot of disposable data was then processed to remove unnecessary data, compressed, and converted to CSV files using `netcap` [30]. Python library `matplotlib` [31] was then utilized to plot the necessary graphs to identify useful features. Python Library `pandas` [32] was then utilized to extract these features from the data. Machine Learning library `scikit learn` [33] was utilized for training ML Models.

3.4.2. Capturing and Processing Data

As mentioned in Section 1.2, capturing and processing data are one of the biggest challenges in this field of research—more so, for networks without the necessary hardware to do so. Consumer routers do not have a dedicated monitor port on them as most commercial switches do. Thus, a low power dedicated device such as a Raspberry Pi can be used for capturing and storing network traffic. A similar network monitor device has been mentioned in [34]. A capture device can be set up in two ways inside a local network

- As a server which listens to packet capture data sent via dedicated programs installed on individual devices in the network. This configuration is not suitable as the approach in this paper focuses on monitoring traffic from IoT devices and programs which capture network traffic and send that to a server that cannot be installed on these devices, as shown in Figure 4.
- As a network bridge that sits between the router and devices. This allows the capture device to trace all the traffic across the network, as shown in Figure 5.
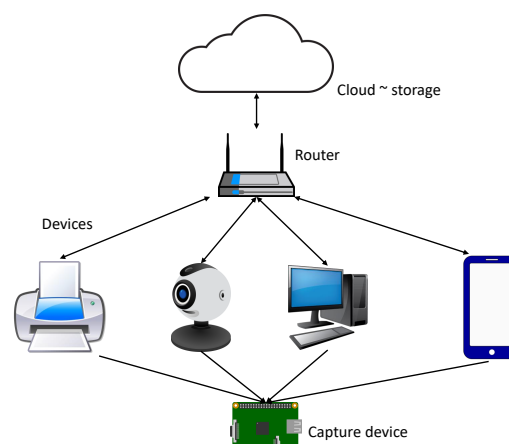

Figure 4. Capture device listening for capture data sent via programs installed on network devices.
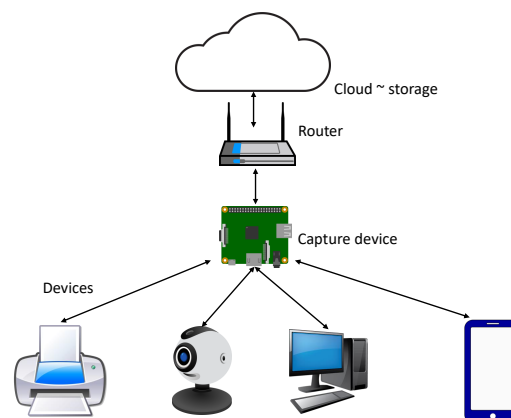

Figure 5. Capture device acting as a network bridge.

3.4.3. Network Dataset

The network data was captured from eight different household networks each containing a multitude of IoT devices. The captured data were then manually cleaned for training the models. Synthetic network traffic for creating anomalies such as internet sweeps and performing DDoS attacks using devices on the network was also generated. Network traffic from well-known malware such as

Mirai whose source code is readily available was also simulated. The features mentioned in the Table 1 were extracted from the data to be used in the machine learning models.

4. Results and Discussion

All of the models were optimized to use the best possible hyperparameters. The dataset is randomly split into training and test sets with an 80:20 ratio. Anomalous data–points are tested directly on the trained model. The results can be visualized in Figure 6. In Tables 3 and 4 containing the results, the Feature column corresponds to the features mentioned in Table 2 and in Table 5 the features column corresponds to the features in Table 1. Results of other techniques anomaly detection techniques on which research has been published haven't been compared since a major part of it relies on the KDD99 Dataset, which is outdated (discussed in [23]); moreover, this paper aims at consumer network traffic instead of commercial network traffic.

Table 3. Performance of features using One Class SVM on features in Table 2.

Feature	Correctly Ident. (Attack)	Correctly Ident. (Normal)	Falsely Ident. (Attack)	Falsely Ident. (Normal)	Avg. DR	Avg. FPR
1	100%	92%	0%	8%	96.00%	4%
2	100%	82.24%	0%	17.76%	91.12%	8.88%
3	100%	94.3%	0%	5.7%	97.10%	2.85%
4	82.35%	70.96%	17.65%	24.04%	72.65%	20.84%
5	80.19%	89.59%	19.81%	10.41%	84.89%	15.11%
6	88.23%	94.12%	11.76%	5.8%	91.17%	8.78%
7	74.36%	94.3%	25.64%	3.7%	84.33%	14.67%
8	100%	75.81%	0%	24.19%	87.90%	12.1%
9	92.4%	77.1%	7.6%	22.9%	84.75%	15.25%
10	100%	78.3%	0%	21.7%	89.18%	10.85%
Avg.	91.753%	84.872%	8.246%	14.42%	87.909%	11.333%

Table 4. Performance of Features using Isolation Forest on features in Table 2.

Feature	Correctly Ident. (Attack)	Correctly Ident. (Normal)	Falsely Ident. (Attack)	Falsely Ident. (Normal)	Avg. DR	Avg. FPR
1	99.6%	96%	0.4%	4%	97.8%	2.2%
2	96.55%	71.69%	3.55%	28.31%	84.12%	15.93%
3	60.12%	90.18%	39.88%	9.92%	77.89%	24.9%
4	57.69%	93.22%	42.31%	6.78%	75.45%	24.54%
5	66.88%	85.06%	33.12%	14.93%	75.97%	24.025%
6	100%	81.25%	0%	19.75%	90.625%	9.87%
7	75%	87.5%	25%	12.5%	81.25%	18.75%
8	80%	89.23%	20%	11.77%	84.615%	15.885%
9	68.14%	90.07%	31.86%	9.83%	79.10%	20.84%
10	100%	81.51%	0%	18.48%	90.75%	9.24%
Avg.	80.398%	86.571%	19.612%	13.627%	83.757%	16.618%

Table 5. Performance of Features using a Hybrid Approach on features in Table 1.

Feature	Correctly Ident. (Attack)	Correctly Ident. (Normal)	Falsely Ident. (Attack)	Falsely Ident. (Normal)	Avg. DR	Avg. FPR
1	46.4%	73.7%	53.6%	26.3%	60.05%	39.95%
2	100.0%	99.3%	0.0%	0.7%	99.65%	0.35%
3	100.0%	99.5%	0.0%	0.5%	99.75%	0.25%
4	100.0%	99.3%	0.0%	0.7%	99.65%	0.35%
5	97.8%	93.4%	2.2%	6.6%	95.60%	4.40%
6	94.2%	94.5%	5.8%	5.5%	94.35%	5.65%
8	92.1%	91.6%	7.9%	8.4%	91.85%	8.15%
Avg.	90.071%	93.042%	9.928%	6.957%	91.55%	8.442%

Anomalous detection rate is defined as the ratio of anomalous data–points classified as anomalies to the total number of anomalous data–points the classifier was tested upon, denoted by DR(a)

$$DR(a) = \frac{\text{No. of anomalous data–points marked positives}}{\text{Total No. of anomalous data–points}} \qquad (1)$$
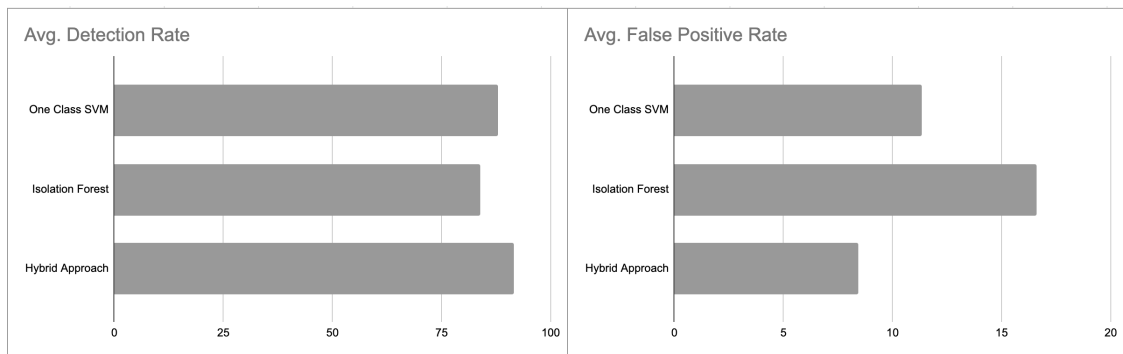
Figure 6. Comparison of Avg. FPR and Avg. detection rates.

Non Anomalous detection rate is defined as the ratio of non-anomalous data–points not classified as anomalies to the test dataset split denoted by `DR(na)`

$$DR(na) = \frac{\text{No. of non-anomalous data–points marked negatives}}{\text{Total No. of non-anomalous data–points}} \qquad (2)$$

Anomalous False positive rate is defined as the ratio of anomalous data–points marked as negatives to the total number of anomalous data–points the classifier was tested upon `FPR(a)`

$$FPR(a) = \frac{\text{No. of anomalous datapoints marked negatives}}{\text{Total No. of anomalous data–points}} \qquad (3)$$

Non Anomalous detection rate is defined as the ratio of non-anomalous data–points classified as anomalies to the test dataset split `FPR(na)`

$$FPR(na) = \frac{\text{No. of non-anomalous data–points marked as positives}}{\text{Total No. of non-anomalous data–points}} \qquad (4)$$

4.1. Isolation Forest

Even with high dimensional data such as the network traffic used in this experiment, isolation forest proves to have sub-par performance amongst the three algorithms compared as seen in Table 3. Its average false positive rate is the highest (16.618%) amongst the three algorithms used. The average detection rate is also the lowest (83.757%). It is worth noting that this approach identifies normal traffic more accurately than OCSVM, but the other inaccuracies outweigh this algorithm's usefulness. The subpar performance might be due to imbalanced classes or being an unsupervised model which was not provided labeled data for training.

4.2. One-Class Support Vector Machine (OCSVM)

Results in Table 3 clearly show that OCSVM outperforms an unsupervised model such as Isolation Forest, by having a better detection rate 87.909% as compared to 83.757% by isolation forest, not only detection rates but also a false positive rate for OCSVM is 11.333% as compared to 16.618%, thus showing that it certainly is superior to Isolation Forest in terms of anomaly detection inside consumer network datasets. Like Isolation forest, optimizing the hyperparameters did help overcome the problems due to imbalanced classes. Although OCSVM requires a small sample size to train the model and proves to be accurate in most cases. In [35], the only downside is the higher than an acceptable false-positive rate, which the prosed hybrid approach surpasses with a sub 10% false-positive rate of 8.442%. This is most likely because raw network traffic data cannot simply be classified as anomalous and normal due to a lot of factors being in play.

4.3. Proposed Hybrid Approach

The hybrid approach not only has the best average detection rate (91.55% as compared to 83.757% and 87.909%) but also has a lower false-positive rate than One-Class Support Vector Machine and Isolation Forest (8.442% as compared to 11.333% and 16.618%), which shows that utilizing entropy values for classification using OCSVM provides better results for detecting network anomalies inside consumer networks than using common ML models.

4.4. Detecting Attacks

Consumer networks are not usually on the receiving side of DDoS Attacks since they are not lucrative targets for cybercriminals, but devices inside a consumer network acting as zombies for a DDoS attack can be identified since they would be sending a lot of packets to the same IP or Port compared to usual. Alongside this, the deviation from usual traffic by being controlled by a botnet (device communicates with the Command and Control Server periodically), and more than the usual amount of traffic of specific kind on a certain port can be used to detect a zero-day attack exploiting a vulnerable service running on a certain port. This way, the proposed method helps detect attacks inside consumer networks.

5. Conclusions

As this research demonstrates, network anomalies can be detected with high accuracy without requiring large labeled datasets. The models proved to accurately detect common network anomalies such as DDoS attacks and port scans being performed by network devices. Other "Novelties" in network traffic were also flagged, which demonstrates the ability to detect Zero-Day attacks on or using consumer devices. The proposed hybrid approach seems to be the forerunner for practical purposes due to the high detection rate, low false-positive rates, and the fact that there is no need for extensive labeled datasets that are unavailable for consumer networks.

As future works, a deep learning approach using Deep Auto Encoder Networks might also prove effective since the semi-supervised ML model OCSVM has also shown promising results in this research. Combining other models with different trade-offs offers a promising direction for future research for securing the next generation of IoT devices.

References

1. Kolias, C.; Kambourakis, G.; Stavrou, A.; Voas, J. DDoS in the IoT: Mirai and other botnets. *Computer* **2017**, *50*, 80–84, doi:10.1109/MC.2017.201.
2. Sharma, V.; You, I.; Yim, K.; Chen, R.; Cho, J.H. BRIoT: Behavior Rule Specification-Based Misbehavior Detection for IoT-Embedded Cyber-Physical Systems. *IEEE Access* **2019**, *7*, 118556–118580.
3. Sharma, V.; Lee, K.; Kwon, S.; Kim, J.; Park, H.; Yim, K.; Lee, S.Y. A consensus framework for reliability and mitigation of zero-day attacks in IoT. *Secur. Commun. Netw.* **2017**, *2017*, 4749085.
4. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly Detection: A Survey. *ACM Comput. Surv.* **2009**, *41*, 1–58, doi:10.1145/1541880.1541882.

5. Patcha, A.; Park, J.M. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Comput. Netw.* **2007**, *51*, 3448–3470, doi:10.1016/j.comnet.2007.02.001.

6. García-Teodoro, P.; Díaz-Verdejo, J.; Maciá-Fernández, G.; Vázquez, E. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Comput. Secur.* **2009**, *8*, 18–28, doi:10.1016/j.cose.2008.08.003.

7. Lo, C.H.; Ansari, N. CONSUMER: A Novel Hybrid Intrusion Detection System for Distribution Networks in Smart Grid. *IEEE Trans. Emerg. Top. Comput.* **2013**, *1*, 33–44, doi:10.1109/tetc.2013.2274043.

8. Zhang, Z.; Li, J.; Manikopoulos, C.N.; Jorgenson, J.; Ucles, J. HIDE: A hierarchical network intrusion detection system using statistical preprocessing and neural network classification. In Proceedings of the IEEE Workshop on Information Assurance and Security, West Point, NY, USA, 5–6 June 2001.

9. Manikopoulos, C.; Papavassiliou, S. Network intrusion and fault detection: A statistical anomaly approach. *IEEE Commun. Mag.* **2002**, *40*, 76–82, doi:10.1109/MCOM.2002.1039860.

10. Mahoney, M.; Chan, P. The Learning Rules for Anomaly Detection of Hostile network traffic; In Proceedings of the Third IEEE International Conference on Data Mining, Melbourne, FL, USA, 22–22 November 2004; doi:10.1109/icdm.2003.1250987

11. Wang, K.; Stolfo, S.J. *Anomalous Payload-Based Network Intrusion Detection*; Springer: Berlin/Heidelberg, Germany, 2010; doi:10.1007/978-3-540-30143-1_11.

12. Xiuyao, S.; Mingxi, W.; Jermaine, C.; Ranka, S. Conditional anomaly detection. *IEEE Trans. Knowl. Data Eng.* **2007**, *19*, 631–645, doi:10.1109/TKDE.2007.1009.

13. Chhabra, P.; Scott, C.; Kolaczyk, E.D.; Crovella, M. Distributed spatial anomaly detection. In Proceedings of the IEEE INFOCOM, Phoenix, AZ, USA, 13-18 April 2008; doi:10.1109/INFOCOM.2007.232.

14. Lu, W.; Ghorbani, A.A. Network anomaly detection based on wavelet analysis. *EEURASIP J. Adv. Signal Process.* **2009**, *2009*, 837601, doi:10.1155/2009/837601.

15. Simmross-Wattenberg, F.; Asensio-Pérez, J.I.; Casaseca-De-La-Higuera, P.; Martín-Fernandez, M.; Dimitriadis, I.A.; Alberola-López, C. Anomaly detection in network traffic based on statistical inference and α-stable modeling. *IEEE Trans. Dependable Secur. Comput.* **2011**, *8*, 494–509, doi:10.1109/TDSC.2011.14.

16. Yu, M. A Nonparametric Adaptive Cusum Method In addition, Its Application In Network Anomaly Detection. *Int. J. Adv. Comput. Technol.* **2012**, *4*, 280–288, doi:10.4156/ijact.vol4.issue1.32.

17. Maple, C. Security and privacy in the internet of things. *J. Cyber Policy* **2017**, *2*, 155–184, doi:10.1080/23738871.2017.1366536.

18. Group, B.I.T.A. Internet of Things (IoT) Security and Privacy Recommendations. Available online: https://www.bitag.org/report-internet-of-things-security-privacy-recommendations.php (accessed on 29 May 2020).

19. Sharma, V.; You, I.; Kumar, R. Isma: Intelligent sensing model for anomalies detection in cross platform osns with a case study on iot. *IEEE Access* **2017**, *5*, 3284–3301.

20. Sharma, V.; Kumar, R.; Cheng, W.H.; Atiquzzaman, M.; Srinivasan, K.; Zomaya, A.Y. NHAD: Neuro-fuzzy based Horizontal Anomaly Detection in online social networks. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 2171–2184.

21. Doshi, R.; Apthorpe, N.; Feamster, N. Machine Learning DDoS Detection for Consumer Internet of Things Devices. In Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 24–24 May 2018.

22. KDD99 Dataset. Available online: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed on 29 May 2020).

23. Creech, G.; Hu, J. Generation of a new IDS test dataset: Time to retire the KDD collection. In Proceedings of the IEEE Wireless Communications and Networking Conference, WCNC, Shanghai, China, 7–10 April 2013; doi:10.1109/WCNC.2013.6555301.

24. Bhuyan, M.H.; Bhattacharyya, D.K.; Kalita, J.K. Network anomaly detection: Methods, systems and tools. *IEEE Commun. Surv. Tutorials* **2014**, *6*, 303–336, doi:10.1109/SURV.2013.052213.00046.

25. Navaz, A.; Sangeetha, V.; Prabhadevi, C. Entropy based anomaly detection system to prevent DDoS attacks in cloud. *arXiv* **2013**, arXiv:1308.6745.

26. Arunraj, N.S.; Hable, R.; Fernandes, M.; Leidl, K.; Heigl, M. *Comparison of Supervised , Semi-supervised and Unsupervised Learning Methods in Network Intrusion Detection System (NIDS) Application*; Anwendungen und Konzepte Der Wirtschaftsinformatik (AKWI): Lucerne, Switzerland 2018.

27. Rassam, M.A.; Maarof, M.A.; Zainal, A. Adaptive and online data anomaly detection for wireless sensor systems. *Knowl. Based Syst.* **2014**, *60*, 44–57, doi:10.1016/j.knosys.2014.01.003.

28. Lyu, M.; Sherratt, D.; Sivanathan, A.; Gharakheili, H.H.; Radford, A.; Sivaraman, V. Quantifying the reflective DDoS attack capability of household IoT devices. In Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks, Boston, MA, USA, 18–20 July 2017; pp. 46–51.

29. Wireshark. Available online: https://www.wireshark.org/ (accessed on 29 May 2020).

30. Netcap. Available online: https://github.com/dreadl0ck/netcap (accessed on 29 May 2020).

31. Matplotlib. Available online: https://matplotlib.org (accessed on 29 May 2020).

32. Pandas. Available online: https://pandas.pydata.org (accessed on 29 May 2020).

33. Scikit-Learn. Available online: https://scikit-learn.org (accessed on 29 May 2020).

34. Mukerji, A.; Rothstein, J. Detecting Anomalous Network Application Behavior. US Patent 8,185,953, 22 May 2012.

35. He, X.; Mourot, G.; Maquin, D.; Ragot, J.; Beauseroy, P.; Smolarz, A.; Grall-Maës, E. Multi-task learning with one-class SVM. *Neurocomputing* **2014**, *133*, 416–426, doi:10.1016/j.neucom.2013.12.022.