Intl. Journal on Cyber Situational Awareness, Vol. 3, No. 1, 2018

Threat Detection and Analysis in the Internet of Things using Deep Packet Inspection

Christopher D. McDermott, William Haynes, Andrei V. Petrovksi

School of Computing Science and Digital Media, Robert Gordon University, UK

ABSTRACT

The Internet of Things (IoT) has quickly transitioned from a promising future paradigm to a pervasive everyday reality. Many consumer IoT devices often lack adequate security and are increasingly being leveraged to perform DDoS attacks. To improve situational awareness of such attacks amongst consumers, this paper presents two solutions to the detection of botnet activity within consumer IoT devices and networks. First, a detection model is built using *Term Frequency-Inverse Document Frequency (tf-idf)* and analyses network traffic for semantic structure, highlighting semantic similarities between the captured data and that of a known attack dataset. A similarity score is used to determine if *mirai* attack vectors could be detected in the captured network traffic. Secondly a novel application of Deep Learning is used to develop a detection model based on a *Bidirectional Long Short Term Memory based Recurrent Neural Network (BLSTM-RNN).* The model is evaluated for accuracy and loss when detecting four attack vectors used by the *mirai* botnet. The paper demonstrates that both approaches return good results and offer promise for future research in this area. A labelled dataset was generated as part of this research and has been made available to the research community.

Keyword: Situational Awareness, IoT, Term Frequency-Inverse Document Frequency, tf-idf, Long Short Term Memory Recurrent Neural Network, Botnet, Intrusion Detection, Mirai, DDoS.

1 INTRODUCTION

The Internet of Things (IoT) is expected to usher in an era of increased connectivity, with an estimated 50 billion devices expected to be connected to the Internet by 2020 (Aazam, St-Hilaire, Lung, Lambadaris, & Huh, 2018). At its core, the aim of the IoT is to connect previously unconnected devices to the Internet (Atzori, Iera, & Morabito, 2010), thus creating smart devices capable of collecting, storing and sharing data, without requiring human interaction (Mosenia & Jha, 2017) (McDermott & Petrovski, 2017). Many of these IoT devices are aimed at consumers, who value low cost and ease of deployment over security. These market forces have resulted in IoT manufacturers omitting critical security features, and producing swathes of insecure Internet connected devices, such as IP cameras and Digital Video Recorder (DVR) boxes. Such vulnerabilities and exploits are often derived and epitomised by inherent computational limitations, use of default credentials and insecure protocols. The rapid proliferation of insecure IoT devices and the ease by which attackers can locate them using online services, such as shodan, provides an ever expanding pool of attack resources. By comprising and leveraging multitudes of these vulnerable IoT devices, attackers can now perform large scale attacks such as spamming, phishing and Distributed Denial of Service (DDoS), against resources on the Internet (Moganedi & Mtsweni, 2017). The rise in IoT based DDoS attacks, witnessed in recent years, will likely continue until IoT manufacturers accept responsibility and incorporate security mechanisms into their devices. Until such a time, the IoT has the potential to become the new playground for future cyber attacks and therefore presents a number of challenges. Since an increasing number of DDoS attacks seek to leverage consumer level IoT devices, the issues highlighted previously, coupled with a lack of technical knowledge or awareness of inherent vulnerabilities, by owners of these devices, presents one such problem. This challenge is further compounded by a lack of convenient user interface on many consumer IoT devices, making detection and awareness of attacks in home networks practically impossible for consumers.

To substantiate this issue, a sand boxed botnet environment was created for preliminary research. An IoT IP Camera was successfully infected and leveraged to perform a sequence of DDoS attacks against a selected target.

During the infection process and attacks, the camera did not display any adverse symptoms of infection and continued to function as expected. Remote access to the device was still possible, and performance did not appear to be degraded. Live video streaming continued to be as responsiveness as prior to the attacks, therefore without any clear signs of an infection it was confirmed that, detection or awareness or botnet activity would prove very difficult within consumer networks.

Current methods of botnet detection, as discussed in Section 2.2, largely rely on signature or flow based anomaly intrusion detection. However, the impact and spread of IoT botnets presented in Section 2.1 would suggest these methods are currently not used or are ineffective in preventing botnet activity within the IoT. This could be due to simple code mutations rendering attack signatures obsolete or a lack of protocol support (*NetFlow, Sflow*) within consumer networks and equipment.

This paper presents two solutions to the detection of botnet activity within consumer IoT devices and networks. Detection was performed at the packet level, and focused on text recognition within features, normally discarded by other flow based detection methods. The first utilised a *Term Frequency-Inverse Document Frequency (tf-idf)* detection model to measure and provide a similarity score between real time network traffic and a dataset with known *mirai* attack vectors included. The second method utilised a novel detection model based on a *Deep Bidirectional Long Short Term Memory based Recurrent Neural Network (BLSTM-RNN)*, in conjunction with Word Embedding, for text recognition and conversion. Thus, the main contributions of this paper are:

1. A labelled dataset encompassing botnet activity and DDoS attacks available at https://tinyurl.com/CMcD-Datasets;
2. A detection model which utilises *Term Frequency-Inverse Document Frequency* to provide a similarity score for traffic behavioral analysis;
3. A detection model which utilises a *Deep Bidirectional Long Short Term Memory based Recurrent Neural Network (BLSTM-RNN)* to extract intelligence from textual features, normally discarded by other flow based detection methods.

2 RELATED WORK

For our related works, we shall consider the topics of botnets in the IoT, botnet detection methods, and situational awareness of botnet activity by non-expert users (NEU).

Although botnet activity and detection has been well researched, the existing literature generally focuses on traditional network botnets, rather than specifically focusing on botnets that target the IoT. Conference papers and peer reviewed articles presented below therefore either directly target IoT botnets or could easily be applied to this research area.

2.1 Botnets in the Internet of Things

Some of the most extensive and destructive cyber-attacks deployed on the Internet have been Distributed Denial of Service (DDoS) attacks. Some of the largest DDoS attacks ever recorded occurred in the second half of 2016, fuelled in full or part by the Internet of Things (Akamai, 2017). During this time, attacks of over 100Gbps were up by 140%, with three attacks reaching over 300 Gbps. The severity of the attacks continued in 2017, evidenced in Verisign's annual DDoS Trends report which reported that 82% of recorded DDoS attacks in quarter 4 of 2017 also now employed a multi-vector attack strategy (Verisign, 2017). IoT Botnets are becoming increasingly more sophisticated in their effectiveness and ability to exploit basic security vulnerabilities, and obfuscate their activity (Kolias, Kambourakis, Stavrou, & Voas, 2017). They present *MalwareMustDie* as an example which uses iptables rules to protect its infected devices, whilst *Hajime* utilises fully distributed communications and makes use of the BitTorrent protocol for peer discovery. *BrickerBot* was also presented which leverages SSH default credentials to perform a permanent denial-of-service (PDoS) attack.

One of the most prominent examples of a DDoS attack emanating from the IoT in recent times, is presented in (Jerkins, 2017) (Sinanovic & Mrdovic, 2017). Mirai is a piece of malware that attempts to find and infect IoT devices to establish and propagate a network of robots (botnet) consisting of the infected IoT devices (bots). An attacker (botmaster) then uses a command and control (C&C) server to remotely control the bots, forcing them to participate in DDoS attacks against targets on the Internet. On September 20 2016 the Mirai botnet was used to perform an unprecedented 620 Gbps DDoS attack on security journalist Brian Krebs website krebsonsecurity.com (Brian Krebs, n.d.). Shortly after it was also responsible for a series of additional DDoS attacks peaking at over 1.2 Tbps against French hosting company OVH and DNS provider DYN, who estimated that up to 100 000 infected IoT devices (bots) were involved in the attack. The severity of the DYN attack was sufficient to cause major disruption on the Internet, and render several high profile websites such as GitHub, Twitter, Reddit, Netflix, inaccessible.

2.2 Botnet Detection Methods

As previously stated much of the existing literature on botnet detection generally focuses on traditional network botnets, rather than IoT botnets.
An increasingly popular approach has been the use of Machine Learning Algorithms (MLA) for network traffic analysis and classification. The assumption being that botnets create distinguishable traffic patterns, that can be used to accurately detect botnet activity (Stevanovic & Pedersen, 2014). In many cases traffic analysis was performed at the network level, analysing *flows* of traffic conversations, rather than at the individual packet level. In doing so, the authors in (Bilge, Balzarotti, Robertson, Kirda, & Kruegel, 2012) used a Support Vector Machine, C4.5 decision tree and Random forest classifier to classify malicious and non-malicious in a *NetFlow* dataset, and harness true positive detection rates above 70%. (Zhao et al., 2013) use Decision trees with the Reduced Error Pruning Algorithm (REPTree) and again demonstrated good performance with true positive detection rates above 90%. It should be noted however, that the use of IP addresses as a prominent feature could result in an unbalanced dataset, and effect detection results. (Kirubavathi & Anitha, 2016) consider smaller packet correlation as a way of improving detection accuracy, by extracting additional features, namely *packet ratio*, *initial packet length*, and *bot response time*, and modelling the behaviour of network flows. Flows were classified using Boosted decision tree (AdaBoostM1+J48), Naive Bayesian (NB), and Support Vector Machine (SVM) algorithms, and the detection system tested against three separate datasets. The authors suggest the advantage of the proposed system is its lightweight nature, however this was not substantiated through comparison with alternative detection methods.

(Stevanovic & Pedersen, 2014) compare eight MLAs for classifying botnet traffic, and also compare two scenarios for traffic analysis. In the first scenario flows are monitored in their entirety from start to end, whereas in the second scenario, traffic flows are only observed for a specific time interval and maximum number of packets. They successfully demonstrated that detection rates could be maintained whilst reducing sample sizes to only 10 packets and 60 seconds of monitored flow traffic. (Stevanovic & Pedersen, 2015) extend their work and propose three methods of traffic analysis for botnet detection, utilising a Random Forest classifier on 40 different bot samples, classifying TCP, UDP and DNS communications separately. Classification accuracy for all protocols was above 90%, although balanced classification required a time of window length of 3600 seconds and 1000 packets, which could result in a lower detection rate for attacks with smaller sample metrics. (Yu, Sekar, Seshan, Agarwal, & Xu, 2015) propose *IoTSec,* which utilises a Software Defined Networking

(SDN) approach to enforcing security policies for IoT devices. Whilst the proposal of a crowd-sourced repository of learned attack signatures, could be useful in detecting botnet activity, it relies on NEUs providing this information, which would prove challenging.

The main drawback for many of these approaches is that they analyse traffic flows rather than individual packets, which results in only representative samples of the total traffic, being considered. In addition, with regard to the problem highlighted in Section 1, it is unlikely that consumer routers would have the ability to capture traffic flows using protocols such as *NetFlow* or *sFlow*, therefore many of the approaches may not be easily transferable to IoT botnet detection.

2.3 Situational Awareness of threats in the IoT

Situational Awareness (SA) can be defined as “the state of being aware of circumstances that exist around us, especially those that are particularly relevant to us and which we are interested about” (Onwubiko & Owens, 2012). Applied in a cyber context the author further presents an adapted situational awareness model comprised of four levels where *perception*, deals with evidence gathering of situations in the network. *Comprehension* refers to the analysis of evidence to deduce threat level, type and associated risk. *Projection* deals with predictive measures to address future incidents, and *resolution* deals with controls to repair, recover and resolve network situations (Onwubiko, 2009).

(Evesti, Kanstren, & Frantti, 2017) suggest cyber SA is often recognised as important, but the ability to systematically evaluate and work on it is often limited. They propose a taxonomy to aid decision makers in structuring and reasoning about cyber security awareness in their context. Three essential elements are presented as necessary to achieve cyber situational awareness. *Data gathering*, from firewalls, anti-virus or vulnerability scanners; *Analysis*, through anomaly detection, parsing logs, or metrics; and *Visualisation*, consisting of statistical, historical and real-time presentation of data. (P. A. Legg, 2016) presents the need for greater online awareness and protection for NEUs. The author undertook a study to establish the views of NEUs on personal cyber security and suggests a lack of technical knowledge and ability to explore network communication, results in little or no awareness of security issues. In response to this a security visualisation framework is proposed to support NEUs to engage with network traffic analysis in order to better support their perception and comprehension of cyber security concerns. The work is extended in (P. Legg, 2016) where the visualisation tool is further developed and used to assess participant ability across two case studies involving malware identification and home network monitoring. Participant feedback was positive, although the results were limited since only a single radial visual representation was used, leaving room for future research in the area.

Despite research in this area it is clear from our preliminary research and information presented in Section 2, that it is still difficult for NEUs to be situationally aware of their network environment and accurately detect and respond to threats posed by IoT botnets.

3 DEEP PACKET DETECTION METHODS

As stated in Section 1 many existing detection models are limited in their application to IoT botnet detection in consumer networks, since analysis is often performed on traffic flows rather than individual network packets. In addition, they often rely on flow based protocols such as *NetFlow* or *sFlow* to analyse network traffic. The two models presented in this section address these limitations by performing deep packet inspection, and focusing on text recognition within features, normally discarded by other flow based detection methods. In particular text within the *info* feature of captured packets (see Table 1). Method one presents a model based on *Term Frequency-Inverse Document Frequency (tf-idf)*, and Method two presents a model based on a *Deep Bidirectional Long Short Term Memory based Recurrent Neural Network (BLSTM-RNN).*

Packet	**Info Feature**
Normal	81 - 50451 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
Mirai	62002 - 23 [SYN] Seq=0 Win=57378 Len=0 [ETHERNET FRAME CHECK
UDP	55741 - 65170 Len=512
DNS	Standard query 0x0c9 A nnt1heibflkk.report.McDPhD.org
ACK	56057 > 49714 [ACK] Seq=1 Ack=1 Win=29597 Len=512

Table 1. Attack Packet Structure

3.1 Data Sources

To evaluate our detection models, we required a dataset which contained a mixture of IoT botnet communication, multiple attack vectors and normal IoT device traffic. There are currently no public datasets that fulfilled all three criteria, therefore an experimental set-up was implemented as shown in Figure 1.

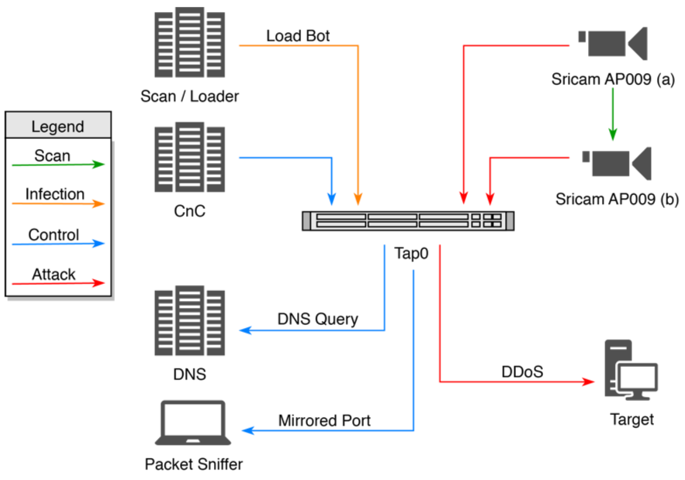


Figure 1. Mirai Botnet Experimental Setup

The *mirai* botnet malware contains ten available attack vectors, which infected IoT devices can utilise to engage in DDoS attacks against targets. For the purpose of our experiments, four attack vectors were chosen, including User Datagram Protocol *(UDP)* flood, Acknowledgement *(ACK)* flood, Domain Name System *(DNS)* flood, and Synchronize *(SYN)* flood attacks, used by *mirai*. Command and control messages between the C&C server and the infected IoT IP camera *(bot)* were also captured, as was normal traffic generated by the camera.

To capture packets and generate the necessary *attack dataset* the *tshark* command line tool and associated input parameters were used. Algorithm 1 shows the process of packet capture, parsing, and storage in a location accessible by the models described in Sections 3.2 and 3.3. The necessary data was captured in a series of five separate captures, which could later be concatenated into a single *attack dataset*. The first capture *(normal.pcap)* consisted of normal IoT device traffic, for a duration of 2 hours and included normal device communication on the network, and also two remote connections to the camera to view the video feed, each of which lasted 5 minutes. *Mobaxterm* was used to create a secure shell *(ssh)* into the C&C server, before executing command *screen ./cnc* from within the *mirai/release* directory, to start the *MYSQL* database. A second remote session was used to telnet and log into the C&C server, ready to issue attack commands to the infected IoT IP camera. A third remote session was used to ssh into the Scan/Loader server, before executing the *./loader* command from within the *mirai/release* directory, to scan the network for available

IoT devices to infect. The initial scanning process and device infection was captured in the second capture *(mirai.pcap)* which also included the infected camera scanning on ports 23 and 2323 for new devices to infect.

The third capture *(udp.pcap)* consisted of a single *(udp)* flood attack, whereby the C&C server issued the attack command, and the infected IoT device flooded its target with bursts of *(udp)* packets for a total period of 60 seconds. The fourth capture *(dns.pcap)* consisted of a single *(dns)* flood attack, whereby the C&C server issued the attack command, and the infected IoT device flooded its target with bursts of *(dns)* packets for a total period of 60 seconds. The fifth capture *(ack.pcap)* consisted of a single *(ack)* flood attack, whereby the C&C server issued the attack command, and the infected IoT device flooded its target with bursts of *(ack)* packets for a total period of 60 seconds.

To perform live traffic analysis in the future, Algorithm 1 can be used to continuously capture and parse data for use with the detection models presented in Sections 3.2 and 3.3. Due to the complexity of training and classification processes, the data pipeline is configured to block future processes (*getLastFileNumber*, *parseToCsv*) from starting until the previous process has completed (*captureNetworkTraffic*). The controlled live input traffic can then be analysed for semantic structure and pattern matched with behaviour found in the *attack dataset*, thus indicating the detection of known attack vectors used by the *mirai* IoT botnet.

3.1.1 Pre-processing and Feature Selection

During the capture and generation of the dataset described in Section 3.1, files were converted from *pcap* to *csv* format (see algorithm 1). Tshark has the ability to capture a vast array of features and protocols, however since one of the proposed detection models utilises machine learning to generate a predictive model, the list was reduced to ensure precision and avoid overfitting the model. Additionally, restricting the number of features reduces the complexity of the model, thus requiring less time and computational power to execute.

Since the two detection models will be applied to consumer IoT devices and networks, the standard seven features used by tshark was selected to maximise coverage and potential adoption. Features *frame.number*, *frame.time*, *ip.src*, *ip.dst*, *ip.proto*, *frame.len* and *col.info* were therefore selected.

Algorithm 1 PACKET capture and parsing

```
1.  procedure parseTshark(saveName, tsharkOut)
2.  directory ← save parsed files
3.  fileNamePrefix ← store parsed file
4.  pcapSuffix ← store pcap file
5.  csvSuffix ← store csv file
6.  tsharkOut ← open tsharkOut in write mode
7.  repeat
8.     /*captureNetworkTraffic*/
9.     saveName ← Join-Path(directory,
    fileNamePrefix, pcapSuffix)
10.    invoke-process ← tshark capture network
    traffic
11. until output to saveName
12. repeat
13.    /*getLastFileNumber*/
14.    for f ← in directory.list() do
15.       if f matches fileNamePrefix then
16.          cutValue ← f.find('.')
17.          if f has no number then
18.             f.append ← everything after
    fileNamePrefix
19.          end if
20.       end if
21.       if not directory then
22.          returnValue ← 0
23.       else
24.          for x ← in length f.append() do
25.             returnValue ← max length
26.          end for
27.       end if
28.    end for
29. until return last file
30. repeat
31.    /*parseToCsv*/
32.    fileNumber ← get last file
33.    invoke-process ← tsharkcall
34. until output to tsharkOut
35. tsharkOut.close()
36. Return parsed tshark output
```

Features *frame.number* and *frame.time* allow each packet in the dataset to be identified by order or timestamp. Features *ip.src* and *ip.dst* provide flow information which could be useful for identifying patterns of traffic flowing through the network. Features *ip.proto* and *frame.len* can be used for pattern matching and classification of packets as normal or anomalous. Finally, the two detection models presented in this paper will assess whether text recognition can be applied to the information provided in the *col.info* feature for botnet detection. The remaining captured features were filtered out during the conversion process, and *csv* files stored for later use by the models described in Sections 3.2 and 3.3.

In order to train and validate the *BLSTM-RNN* detection model in Section 3.3, ground-truth labels *norm, mirai, udp, dns, ack* were assigned to the captured *attack dataset*, ready to be ingested into the detection model.
To evaluate the *td-idf* detection model described in Section 3.2 a second *input dataset* was required. This was generated using Algorithm 1 to capture, parse, and store live network traffic as a series of parsed reference files. For consistency the same five captures were generated, namely *norm, mirai (syn), udp, dns,* and *ack*.

3.2 Method One: Term Frequency-Inverse Document Frequency (td-idf)

This section presents the use of *Term Frequency-Inverse Document Frequency (tf-idf)* as a method of IoT botnet detection. The captured *attack dataset* containing known attack vectors from the *mirai* malware, was compared against the *input dataset* generated from live network traffic. Plain-text documents were analysed for semantic structure to highlight semantically similar documents between the *attack dataset* and *input dataset*. A similarity score was generated and used to determine if *mirai* attack vectors could be detected in the captured live data streams (see algorithm 2).

The *gensim* Python Library was used to tokenise the *attack dataset* and provide a similarity score between live network traffic (*input dataset*) and the *attack dataset* (captured in Section 3.1). Each document in the *attack dataset* was converted from string format and tokenised into a list of tokens using the *word_tokenise* function from *nltk.tokenise* library in the *Natural Language Tool Kit (NLTK)*. A dictionary was then created mapping every tokenised word to an integer. A corpus was created from the dictionary, which listed the number of times each tokenised word occurred in the document. Each document then became a list of tuples where the first number in each tuple was the integer of the tokenised word, and the second number was how many times it appeared in the document. Since the corpus was effectively a tuple it is important to note word order was lost, therefore time series analysis was no longer possible. A Term frequency-inverse document frequency (tf-idf) model was then generated from the corpus, showing the number of documents and tokens. Term Frequency showed how often the word appeared in the document, and Inverse Document Frequency scaled the value by how rare the word was in the corpus. Commonly appearing tokens, but which have little value, were therefore not valued too highly in the similarity measure.

Algorithm 2 tf-idf IoT Botnet Detection

```
1.  procedure similarityComparison(tsharkOut, similarityOut)
2.  captureDirectory ← output location
3.  i.add_watch ← watch capture directory
4.  repeat
5.      /*createSimilarity*/
6.      for f ← in directory.list() do
7.          if f endswith csv then
8.              rawDocuments ← open last tsharkOut in write mode
9.              genDocs ← for w in word_tokenise(text)
10.             for text in rawDocuments
11.             dictionary ← map words to integer dictionary(genDocs)
12.             corpus ← word frequency dictionary.docs2bow(genDocs)
13.             tf-idf ← create tf-idf model tfidfmodel(corpus)
14.             for i ← in corpus do
15.                 s += len(i)
16.             end for
17.             sims ← generate word rarity similarity(tf-idf[corpus])
18.             reader ← read csv input file
19.             query_doc ← generate query doc and convert to tf-idf
20.             for row ← in reader do
21.                 for x ← in range do
22.                     query_doc += word_tokenise(row[x])
23.                 end for
24.             end for
25.             query_doc_bow ← generate query corpus
26.             query_doc_tf-idf ← generate td-idf model
27.             sims[query_doc_tf-idf] ← array of document similarities
28.             sizeOfDoc ← 0
```

```
29.             for pair ← in query_doc_tf-idf do
30.                 sizeOfDoc += 1
31.             end for
32.         end if
33.     end for
34. until output to similarityOut
35. rawDocuments.close()
36. Return overall document similiarity
```

In order to determine if *mirai* attack vectors can be detected in the live network traffic, data streams were captured as described in Section 3.1 and stored as an *input dataset,* in a defined directory. The tokenisation process previously described was repeated on the *input dataset* to create a series of tokenised query documents, where again a dictionary was created mapping every tokenised word to an integer, and a corpus generated from the dictionary. A tf-idf model was again generated from the corpus, showing the number of documents and tokens in the *input dataset*.

The tokenised documents within each dataset were compared and analysed to highlight semantically similar documents between the *attack dataset* and *input dataset*. Each match between the datasets, returned a score value, and were summed to generate an overall similarity score between the two datasets. Thus, IoT Botnet detection is achieved by recognising pattern behaviour in live network traffic which matches known attack vectors found in the *mirai* malware.

3.3 Method Two: Bidirectional Long Short Term Memory Recurrent Neural Network (BLSTM-RNN)

The developed model for method two uses a novel application of a Deep Bidirectional Long Short Term Memory based Recurrent Neural Network (BLSTM-RNN), in conjunction with Word Embedding, to convert string data found in captured packets, into a format usable by the BLSTM-RNN.

The dataset used in our experiments was generated from the experimental set-up described in Section 3.1. It consists of Mirai botnet traffic such as *Scan*, *Infect*, *Control* and *Attack* traffic as described in Section 3.1 and normal IoT IP Camera traffic generated in our experimental set-up. The dataset included features *No.*, *Time, Source, Destination, Protocol, Length,* and overall payload information in the *Info* feature. Some features such as *No.* and *Time* did not provide much scope for data analysis so were later removed.

Majority of the captured information resided in the *info* feature, as shown in Table 1 therefore a model was required which could read and understand the text presented in this feature.

Artificial Neural Network(ANN) and more complex versions of Recurrent Neural Networks(RNN) such as Long Short Term Memory (LSTM) only work with numerical values. However (Ray, Rajeswar, & Chaud, 2015) demonstrated that a Deep Bidirectional Long Short Term Memory based RNN (BLSTM-RNN) can be used which provides promising results for text recognition. (Wang, Qian, Soong, He, & Zhao, 2015) further demonstrated this potential when a BLSTM-RNN was used in conjunction with Word Embedding, in such a way phrases and vocabulary were mapped to vectors or real numbers and proved to be an effective method for modelling and predicting sequential text.

Artificial Neural Network(ANN) and more complex versions of Recurrent Neural Networks(RNN) such as Long Short Term Memory (LSTM) only work with numerical values. However (Ray, Rajeswar, & Chaud, 2015) demonstrated that a Deep Bidirectional Long Short Term Memory based RNN (BLSTM-RNN) can be used which provides promising results for text recognition.

Algorithm 3 BLSTM IoT Botnet Detection

```
1.  procedure dataProcessing(attack dataset)
2.  path ← attack dataset location
3.  allFiles ← open pattern matched csv files in
    write mode
4.  frame ← define two dimensional labelled
    data structure
5.  unitToDrop ← 25%
6.  repeat
7.     /*create concatenated dataset*/
8.     for i ← in allFiles do
9.        df ← read files
10.       list_ ← append(df) read files
11.    end for
12. until files concatenated into dataset
13. dataset ← concatenated (list_)
14. repeat
15.    /*integer encode dataset*/
16.    for d ← in dataset.values do
17.       encoded_docs ← tokenise words
18.       dict ← create dictionary of
```

```
    encoded_docs
19.         array ← map indices of dict
20.         if array length != 25 then
21.             invoke-process ← pad array == 25
22.         end if
23.     end for
24. until data tokenised and integer encoded
25. padded_docs ← array of tokenised and padded text
26. dataset.dropna ← split dataset based on unitToDrop
27. repeat
28.     /*train and evaluate model*/
29.     model.compile ← (loss == mae, optimizer == adam)
30.     for i ← in epochs do
31.         reshape ← Training and Test to 3 dimension
32.         model.evaluate ← Accuracy and Loss
33.     end for
34. until trainingDataset and testDataset are reshaped
35. Return Loss, ValLoss, Acc, ValAcc
```

(Wang, Qian, Soong, He, & Zhao, 2015) further demonstrated this potential when a BLSTM-RNN was used in conjunction with Word Embedding, in such a way phrases and vocabulary were mapped to vectors or real numbers and proved to be an effective method for modelling and predicting sequential text.

Motivated by this potential, this section presents a detection algorithm and model, which is applied to botnet detection in the IoT. Since the information provided in the *Info* feature of the dataset follows a sequence, we implemented our approach by first converting each letter into a tokenized and integer encoded format. A dictionary of all tokenized words and their index within the *Info* feature was created and text replaced with its corresponding index number. In order to understand each attack type, it was important to maintain the sequence order of the indices, therefore an array of the indices was created.

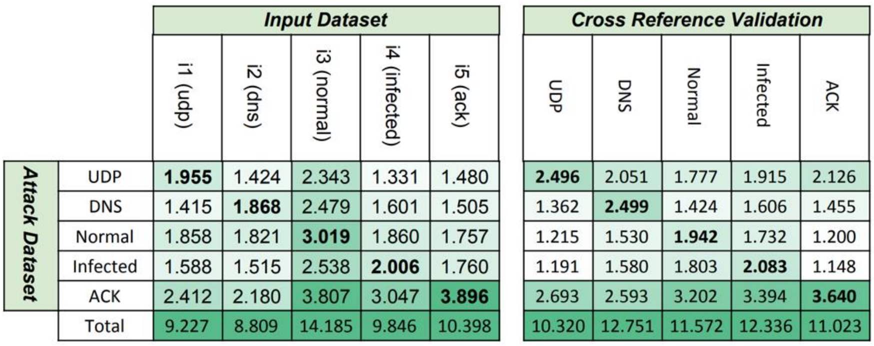
Since attacks are often closely coupled to the protocol used and the length of the captured packet, the *Protocol* and *Length* features also required to be included in the array. Word Embedding was again used to convert and create a dictionary of all tokenized protocols and their index. These were then added, along with the *Length* feature, which was already an integer, to the array. Labels identifying each type of captured packets were mapped from string to integer ('norm': 0,'mirai':1,'udp': 2, 'dns':3, 'ack':4,'normal':5), and also injected into the array. To simplify this process, we used the *Keras* library with a wrapper API around *Theano* and *Tensorflow*. The Keras *one_hot* function was used to convert strings into indices, form a 2-dimension list and create a dictionary at the same time.

Finally, since deep neural networks require arrays to be of equal length, we needed to find the maximum length of a sentence within the *Info* feature and pad all the arrays with 0 to be equal to the maximum length of 25.
After processing the dataset, it was split into *training* and *test* datasets and reshaped into 3 dimensions, the format required for LSTM layer (see algorithm algorithm 3.)

4 RESULTS

To test the *tf-idf* detection model data captures generated in Section 3.1 were presented to the model. Similarity scores were recorded, and a heat map generated as shown in Figure 2, where lighter cells represent a low similarity score, whilst darker cells represent a stronger similarity match between captures. Data in the first five columns represent a series of data captures from the *input dataset*, which include associated attack vectors.

When compared against captures containing the same attack vector in the *attack dataset* a small amount of differentiation is evident. Although small, the differentiation highlights a few trends which are indicative of behavioral analysis occurring. Similarity scores highlighted in bold show each capture in the *input dataset* receives the highest similarity score when compared against a capture with the corresponding attack vector in the *attack dataset*. This suggests that attack vectors in the live data streams are being accurately matched with known attack vectors in the *attack dataset,* which is very promising. It is also worth noting that capture *i3 (normal)* received the highest total similarity score across all captures in the *attack dataset*. This is further confirmation of behavioral analysis taking place, *since normal* packets would be present in all five captures, again showing they are correctly being matched between the datasets.



		Input Dataset					Cross Reference Validation				
		i1 (udp)	i2 (dns)	i3 (normal)	i4 (infected)	i5 (ack)	UDP	DNS	Normal	Infected	ACK
Attack Dataset	UDP	**1.955**	1.424	2.343	1.331	1.480	**2.496**	2.051	1.777	1.915	2.126
	DNS	1.415	**1.868**	2.479	1.601	1.505	1.362	**2.499**	1.424	1.606	1.455
	Normal	1.858	1.821	**3.019**	1.860	1.757	1.215	1.530	**1.942**	1.732	1.200
	Infected	1.588	1.515	2.538	**2.006**	1.760	1.191	1.580	1.803	**2.083**	1.148
	ACK	2.412	2.180	3.807	3.047	**3.896**	2.693	2.593	3.202	3.394	**3.640**
	Total	9.227	8.809	14.185	9.846	10.398	10.320	12.751	11.572	12.336	11.023

Figure 2. Similarity Score Heat Map

To validate the accuracy of the model, the five captures in the *input dataset* were duplicated and presented back to the model for cross reference validation. Similarity scores highlighted bold in columns six to ten clearly show that when two identical captures are compared against each other, the highest similarity score is returned.

A final observation is that the *ack* attack vector returned the highest similarity score consistently across all tests. This is possibly due to how the *(tf-idf)* model calculates the frequency and value of words in the corpus. Term Frequency is used to show how often the word appeared in the document, and Inverse Document Frequency scaled the value by how rare the word was in the corpus. Commonly appearing tokens, but which have little value, are valued lower in the similarity measure. In the case of *ack* packets the attack vector generated a larger number of packets when performing the attack, therefore appearing more frequently in the captures. In addition, as shown in Table 1, *ack* and *normal* packets contain similar string value items such as *Ack*, *Win* and *Len*. Although some items such as *Win* and *Len* are also shared with other packet types, only *ack* and *normal* packets contain the *ack* string value. Since *normal* packets appear in every capture and therefore most frequently in the overall corpus, when matched with *ack* packets, *ack* is likely given a higher *IDF* value. This likely accounts for why these packet types and capture files return the highest similarity scores.

To test the *BLSTM-RNN* detection model a series of four experiments were performed. For *Experiment 1* each attack type was split between *train* and *validate*, presented to the model and trained over a total of 20 iterations. The mean accuracy and loss metrics for each attack were measured and are presented in Table 2.

As can be seen from the results, the model returned high accuracy and prediction for *mirai*, *udp*, and *dns* attack types. However, returned less favourable results for *ack* attacks, despite this attack having the highest number of samples. This was possibly due to the nature and complexity of information in the *info* feature, as seen in Table 3, where the sequence numbers in each *ack* packet changed.

Despite this, a pattern can however be seen on rows one and two, where sequence numbers *(59693-41058, 41058-59693)* of contiguous packets were clearly linked, and packet size and Length were consistent. Unfortunately, some packets appeared out of sync as can been in rows three and four, and possibly resulted in the detection model not recognising this pattern, contributing to the lower detection rate, and significantly higher loss metric. By contrast, although the *mirai* captured packets in Table 1 appear to be equally complex, the information in the *info* feature, remained largely the same, possibly aiding better detection.

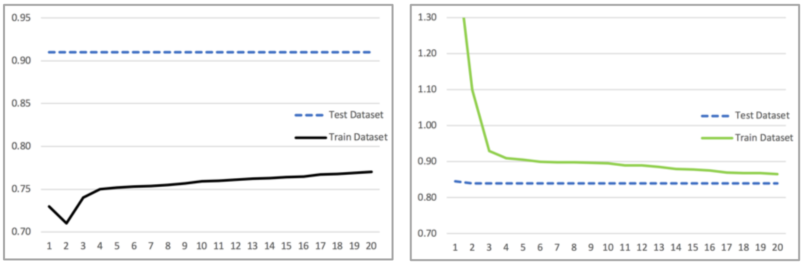
Since multi-vector DDoS attacks were highlighted as being a growing issue in Section 2.1, *Experiment 2* consisted of *norm, mirai, udp, dns,* and *ack* captures being concatenated to form a multi-vector attack scenario. Results on row 5 of Table 2 show the negative impact of the *ack* attack on the overall detection accuracy and particularly loss metrics. To validate this observation, *Experiment 3* consisted of *norm, mirai, udp,* and *dns* captures being concatenated to form a multi-vector attack scenario, minus the *ack* attack. Results on row 6 of Table 2 demonstrate that once the *ack* attack is removed, overall detection accuracy and prediction of the model improves.

Packet	**Train**	**Validate**	**Mean Accuracy**	**Mean Loss**
Mirai	387060	208418	99.998992	0.000809
UDP	391002	210540	98.582144	0.125630
ACK	411384	221515	93.765198	0.858700
DNS	391622	210874	98.488289	0.116453
Multi-Vector (with ACK)	419887	226094	91.951002	0.841303
Multi-Vector (without ACK)	395564	212996	97.521033	0.115293
Multi-Vector (with three ACK)	468534	252289	92.243513	0.161890

Table 2. BLSTM-RNN Detection Accuracy and Loss

Packet	**Info Feature**
ACK	59693 - 41058 [ACK] Seq=1 Ack=1 Win=29597 Len=512
ACK	41058 - 59693 [ACK] Seq=1 Ack=1 Win=29597 Len=0
ACK	28029 - 45060 [ACK] Seq=1 Ack=1 Win=29597 Len=512
ACK	56493 - 64047 [ACK] Seq=1 Ack=1 Win=29597 Len=512

Table 3. ACK Packet Structure and Sequencing



(a) BLSTM Accuracy (b) BLSTM Loss

Figure 3. BLSTM-RNN Accuracy and Loss

A final validation of this observation was conducted in *Experiment 4* which consisted of three *ack* attacks performed during the same time frame, increasing the total sample size of *ack* attacks, in order to observe the variation in accuracy and prediction. Row 7 of Table 2 demonstrations an increase in sample size, improves the overall validation accuracy slightly to 92%, with a significantly better loss metric, when compared to *Experiment 2*. This suggests the model could better predict attack traffic, when presented with a larger sample size, since the sequencing pattern of *ack* packets shown in Table 3, may now be detected by the model.

5 CONCLUSION

This paper presents two solutions to the detection of botnet activity within consumer IoT devices and networks. Detection was performed at the packet level, and focused on text recognition within features, normally discarded by other flow based detection methods.

The *Term Frequency-Inverse Document Frequency (tf-idf)* detection model demonstrated the effectiveness of using text recognition for detecting attack vectors associated with IoT botnets. The results presented in Figure 2 are encouraging and demonstrate how a similarity score could be used for traffic behavioral analysis. The paper also presents a second detection method based on a Deep *Bidirectional Long Short Term Memory based Recurrent Neural Network (BLSTM-RNN)*, in conjunction with Word Embedding, for text recognition and conversion. The bidirectional nature of the model utilised contextual information from both past and future and demonstrated strong accuracy and loss metrics for our captured datasets. Both models demonstrated that by focusing detection at the packet level and using text recognition on features often discarded by specification or flow based detection methods, botnet detection in consumer IoT environments could be improved.

5.1 Limitations

Although results presented in this paper are very encouraging and provide a platform for further research in the area, some limitations are evident. Overall study design was limited by a lack of suitable public datasets in this research area. A new dataset was generated during this research; however, the detection methods have only been tested against a single malware type (*mirai*) and camera model. Further research is required to test how well the models translate to other types of malware found in the IoT.

6 FUTURE WORK

Several avenues for future research have been identified. Firstly, this research could be expanded to demonstrate the ability of our developed models to detect new mutated variants of the *mirai* botnet. The models could also be tested against other types of malware not found within the *mirai* family. Interestingly this paper found the *ack* attack vector metrics to be less favourable in the *BLSTM-RNN* model but returned the highest similarity scores in the *tf-idf* model. A future research direction could be to explore if the two models could be combined to develop a detection engine capable of detecting a wider range of attack vectors associated with IoT botnets.

Finally having successfully demonstrated a solution to the detection problem presented in Section 1, we will also further investigate ways to improve situational awareness of botnet activity within the IoT. By helping consumers become aware when their device is infected, we hope to raise awareness of the inherent vulnerabilities, and aid them to make better choices in the future, with regard to procurement, and operation of such devices.

7 REFERENCES

Aazam, M., St-Hilaire, M., Lung, C.-H., Lambadaris, I., & Huh, E.-N. (2018). IoT Resource Estimation Challenges andModeling in Fog. In A. M. Rahmani (Ed.), *Fog Computing in the Internet of Things: Intelligence at the Edge* (pp. 17–31). Springer International Publishing AG.

Akamai. (2017). *Threat Advisory: Internet of Things and the Rise of 300 Gbps DDoS Attacks | Akamai*. Retrieved from https://www.akamai.com/us/en/multimedia/documents/social/q4-state-of-the-internet-security-spotlight-iot-rise-of-300-gbp-ddos-attacks.pdf

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, *54*(15), 2787–2805. https://doi.org/10.1016/j.comnet.2010.05.010

Bilge, L., Balzarotti, D., Robertson, W., Kirda, E., & Kruegel, C. (2012). DISCLOSURE : Detecting Botnet Command and Control Servers Through Large-Scale NetFlow Analysis. In *28th Annual Computer Security Applications Conference* (pp. 129–138).

Brian Krebs. (n.d.). KrebsOnSecurity Hit With Record DDoS. Retrieved January 10, 2018, from https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/

Evesti, A., Kanstren, T., & Frantti, T. (2017). Cybersecurity Situational Awareness Taxonomy. In *International Conference On Cyber Situational Awareness, Data Analytics And Assessment, Cyber SA 2017* (pp. 1–8). https://doi.org/10.1109/CyberSA.2017.8073386

Jerkins, J. A. (2017). Motivating a market or regulatory solution to IoT insecurity with the Mirai botnet code. In *IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)* (pp. 1–5). https://doi.org/10.1109/CCWC.2017.7868464

Kirubavathi, G., & Anitha, R. (2016). Botnet detection via mining of traffic flow characteristics. *Computers and Electrical Engineering*, *50*, 91–101. https://doi.org/10.1016/j.compeleceng.2016.01.012

Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. (2017). DDoS in the IoT: Mirai and other Botnets. *Computer*, *50*(7), 80–84. https://doi.org/10.1109/MC.2017.201

Legg, P. (2016). Visual Analytics for Non-Expert Users in Cyber Situation Awareness. *International Journal on Cyber Situational Awareness*, *1*(1), 54–73. https://doi.org/10.22619/IJCSA.2016.100103

Legg, P. A. (2016). Enhancing cyber situation awareness for Non-Expert Users using visual analytics. *International Conference on Cyber Situational Awareness, Data Analytics and Assessment, CyberSA 2016*, (Figure 1), 1–8. https://doi.org/10.1109/CyberSA.2016.7503278

McDermott, C. D., & Petrovski, A. V. (2017). Investigation Of Computational Intelligence Techniques for Intrusion Detection in Wireles Sensor Networks. *International Journal of Computer Networks & Communications (IJCNC)*, *9*(4), 45–56. https://doi.org/10.5121/ijcnc.2017.9404

Moganedi, S., & Mtsweni, J. (2017). Beyond the Convenience of the Internet of Things : Security and Privacy Concerns. In *IST-Africa Week Conference* (pp. 1–10).

Mosenia, A., & Jha, N. K. (2017). A Comprehensive Study of Security of Internet-of-Things. *IEEE Transactions on Emerging Topics in Computing*, *5*(4), 586–602.

Onwubiko, C. (2009). Functional Requirements of Situational Awareness in Computer Network Security. *2009 IEEE International Conference on Intelligence and Security Informatics*, 209–213. https://doi.org/10.1109/ISI.2009.5137305

Onwubiko, C., & Owens, T. (2012). Review of Situational Awareness for Computer Network Defense. In *Situational Awareness in Computer Network Defense: Principles, Methods and Applications*.

Ray, A., Rajeswar, S., & Chaud, S. (2015). Text Recognition using Deep BLSTM Networks. In *Eighth International Conference on Advances in Pattern Recognition (ICAPR)*. https://doi.org/10.1109/ICAPR.2015.7050699

Sinanovic, H., & Mrdovic, S. (2017). Analysis of Mirai malicious software. *25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 1–5. https://doi.org/10.23919/SOFTCOM.2017.8115504

Stevanovic, M., & Pedersen, J. M. (2014). An efficient flow-based botnet detection using supervised machine learning. In *2014 International Conference on Computing, Networking and Communications (ICNC)* (pp. 797–801).

Stevanovic, M., & Pedersen, J. M. (2015). An analysis of network traffic classification for botnet detection. In *2015 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)* (pp. 1–8). https://doi.org/10.1109/CyberSA.2015.7361120

Verisign. (2017). *DISTRIBUTED DENIAL OF SERVICE TRENDS REPORT* (Vol. 4).

Wang, P., Qian, Y., Soong, F. K., He, L., & Zhao, H. (2015). A unified Tagging Solution: Bidirectional LSTM Recurrent Neural Network with Word Embedding. *arXiv E-Prints*.

Yu, T., Sekar, V., Seshan, S., Agarwal, Y., & Xu, C. (2015). 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* (p. 5:1--5:7).

Zhao, D., Traore, I., Sayed, B., Lu, W., Saad, S., Ghorbani, A., & Garant, D. (2013). Botnet detection based on traffic behavior analysis and flow intervals. *Computers & Security*, *39*, 2–16. https://doi.org/10.1016/j.cose.2013.04.007

KEY TERMS

Situational Awareness - End-user perception of issues relating to cybersecurity within their environment.

Internet of Things (IoT) - Extensive network of connected 'things' , capable of sensing the surrounding environment and interacting with other devices, to aid real-time monitoring and decision making

Term Frequency-Inverse Document Frequency (tf-idf) – Information retrieval technique used to measure the importance of a word or term to a document.

Long Short Term Memory Recurrent Neural Network – Type of Machine Learning well suited to classifying, processing and making predictions based on time series data.

BIOGRAPHICAL NOTES

Christopher D. McDermott is a Lecturer in the School of Computing Science and Digital Media at Robert Gordon University, Scotland. Following a successful period in industry he is now a PhD candidate and a

member of the Security and Privacy research group. His current research interests include, but are not limited to, Network Security, Cyber Situational Awareness, IoT Security, Intrusion Detection, Information Security and Privacy, and Blockchain technologies.

William Haynes works for National Grid having successfully completed his BSc (Hons) Computer Science at Robert Gordon University. His current research interests include, but are not limited to, Penetration Testing, Cyber Security, IoT Security.

Andrei Petrovski is a Reader in Computational Systems in the School of Computing Science and Digital Media at Robert Gordon University, Scotland. He completed his MSc in Electrical Engineering at Samara State University in Russia, and his PhD in computing at Robert Gordon University. His research interests include computational modelling, optimisation and decision support, practical applications of computer-assisted measurements, fault diagnosis and predictive control.

REFERENCE

Reference to this paper should be made as follows: McDermott, C. D., Haynes, W. & Petrovski, A. V. (2018). Threat Detection and Analysis in the Internet of Things using Deep Packet Inspection. *International Journal on Cyber Situational Awareness*, Vol. 3, No. 1, pp. 61-83.