Anomaly Detection Models for Smart Home Security

Sowmya Ramapatruni, Sandeep Nair Narayanan, Sudip Mittal, Anupam Joshi, and Karuna Joshi
Computer Science and Engineering,
University of Maryland, Baltimore County, Baltimore, Maryland 21227
Email: {sowmya1, sand7, smittal1, joshi, kjoshi1}@umbc.edu

***Abstract*—Recent years have seen significant growth in the adoption of smart homes devices. These devices provide convenience, security, and energy efficiency to users. For example, smart security cameras can detect unauthorized movements, and smoke sensors can detect potential fire accidents. However, many recent examples have shown that they open up a new cyber threat surface. There have been several recent examples of smart devices being hacked for privacy violations and also misused so as to perform DDoS attacks. In this paper, we explore the application of big data and machine learning to identify anomalous activities that can occur in a smart home environment. A Hidden Markov Model (HMM) is trained on network level sensor data, created from a test bed with multiple sensors and smart devices. The generated HMM model is shown to achieve an accuracy of 97% in identifying potential anomalies that indicate attacks. We present our approach to build this model and compare with other techniques available in the literature.**

I. Introduction

A smart home environment consists of advanced automation systems like voice assistants, thermostats, lighting, motion sensors, cameras, doorbells, locks etc. which provide inhabitants with sophisticated monitoring and control over various functions of the home[1]. These devices can be monitored and controlled remotely through the internet, making smart home an important application of the Internet of Things (IoT). Towards the end of 2018, an estimated 45 million houses were equipped with IoT devices and sensors [1]. This increased adoption of smart devices also brings forth threats to user security and privacy. Cyber attacks and intrusion of user's privacy are on a rise over the past few years. In a recent incident, smart thermostats in Finland were attacked causing the residents to suffer harsh cold conditions in the winter [19]. The problem will be further exacerbated as the number of devices per person is set to reach new highs in a few years [13].

Various data streams and corresponding events generated by these sensors and smart devices provide continuous information about activities occurring in a house. This "big data" can also help in establishing the day-to-day 'normal' in a house [8] [7] [12]. Events, or the corresponding network traffic data to and from these devices, that deviates from this normal can be an indicator of an attack. These 'anomalous events' once detected can help protect a home IoT environment.

Anomaly detection is a challenging problem and the varied threat landscape of a smart home add to its complexity. The number of devices connected to the internet in a home provides a variety of attack surfaces that can be leveraged by a hacker to invade a resident's security and privacy. Many of the devices available in the market today do not adhere to security practices and standards that one would find on personal computers. At the price point and mass market appeal through ease of use these devices desire, security is not the primary concern for various device manufacturers. Many attackers have leveraged vulnerabilities in various home IoT systems, such as a fixed well known password for telnet. Botnets like, Mirai [11], Reaper [14], etc. have leveraged known vulnerabilities in smart home devices.

To achieve the goal of an efficient and secure system we need to develop big data and machine learning based anomaly detection models. These detection models, operating in a smart home environment should be able to adapt to ever-growing threat landscape, new attack scenarios, and the growing number of smart devices.

In this paper, we aim to use machine learning algorithms like Hidden Markov Models to learn normal household activities in a smart home. We also create a new data set using a smart home environment. Our smart home test bed contains multiple IoT devices like smart plugs[2], wireless sensor tags[3], Nest Protect[4], and Google Mini voice assistant[5]. The data from all these devices is collected and aggregated in a centralized location. The HMM is trained with normal data collected during the course of our experiments. We use the HMM to perform tests to see if the model can detect various anomalous scenarios. In this paper, our first contribution is the Collection, aggregation, and characterization of data from multiple sensors deployed in a smart home environment. Our next contribution is to model the typical behaviors of this smart home environment using Hidden Markov models that can be used for detecting abnormal behaviors. We also developed several anomalous scenarios specific to our environment, our model achieved an accuracy of 97 % in identifying them.

The structure of the remaining paper is as follows: Section II describes some related work and Section III briefly explains the deployment of IoT sensors to gather data on household activities. Section IV, V discusses data modeling and Hidden

[1]http://smarthomeenergy.co.uk/whatsmart-home

[2]https://www.tp-link.com/ca/products/details/cat-5258_HS105.html

[3]http://wirelesstag.net/

[4]https://nest.com/smoke-co-alarm/overview/

[5]https://store.google.com/us/product/google_home_mini

Markov Model training. Section VI presents results and also outlines various simulated anomalous scenarios that are then detected by the trained HMM model. Finally, section VII concludes the research with suggestions for some future work.

II. Related Work

In this section, we discuss some related work to our approach and discuss how our approach is different. Novak et al. [16] outline a technique for anomaly detection in user behaviors for a smart home. The main aspect of their work is to identify unusual short/long activities that happen in a home environment. They used a neural network (self-organizing maps) to identify various anomalous activities. Unlike our approach, they used only the binary behavioral output of limited sensors placed in a home environment and their detection technique is based on the duration of activities which can lead to many false positives. Kanev et al. [10] worked on a similar problem of anomaly detection in home automation systems. They investigated various networks attacks like DDoS and man-in-the-middle in smart home sensors. Neural networks were used to recognize various anomalies in the network by considering the inbound and outbound packets of each sensor per unit of time. Arrignton et al. [2] presented an approach for a real-world simulation service called Open Simulation (OpenSim). It uses IoT capable objects to detect behavioral-based anomalies within a simulated smart home. This work is limited to only simulated data and does not include real-world data.

In our work, we use Hidden Markov Models (HMM) for detecting anomalies. Srivastava et al. [21] proposed a technique to do credit fraud detection using HMM. They trained an HMM using normal/legitimate credit transactions and identified a threshold of acceptance probability using HMM. Any sequence of the transaction which is not accepted by the trained model with high probability is considered fraudulent. Nguyen et al. [15] and Duong et al. [3] introduced the possibility of learning and identifying various daily activities in a smart home environment using a variant of Hidden Markov Models. They monitored various home activities using some cameras installed in a house. The pre-processed data is later labeled into activities and these activities are modeled using a HMM. The trained HMM is used to identify anomalous activities.

Joshi et al. [9] created Hidden Markov Models to detect normal and abnormal network traffic. They claimed that they achieved 79% accuracy in detecting anomalous network traffic. The main drawback with this approach is the data-set used for various experiments. KDD Cup data-set is about a decade old, and no longer adequate for any current study [20].

Yi Zeng et at. [22] proposed machine learning based intrusion detection methods to automatically detect intruders in Vehicular Ad-hoc networks. They used ANNs and SVMs for implementing their approach. Meikang Qiu et al. [17] presented a novel energy-aware fault tolerance mechanism for wireless sensor networks called Informer Homed Routing. They showed that their mechanism consumes less energy than existing protocols like LEACH and DHR. Keke Gai et al. [4] [5] presented enhanced encryption techniques for ensuring user privacy by encrypting the data that is transmitted among the internet of things and mobile devices. They also [6] proposed a consortium blockchain-oriented approach to solve the problem of privacy leakage without restricting trading functions.

III. Data Collection

For this research, we created a smart home test environment by deploying IoT sensors in a home. Data is aggregated from various IoT sensors deployed and it is further pre-processed for using with HMM models. Figure-1 depicts the layout of smart home and placement of various sensors. It is a single bedroom dwelling with a living room, bedroom, and a kitchen. Data is collected for a total period of 3 weeks for experiments. The sensors and smart devices that are used in the smart home are smart plugs, wireless sensor tags, Nest Protect, and Google Mini voice assistant as shown in Figure 2. During data collection, we consider a single person as the sole occupant in the smart home.

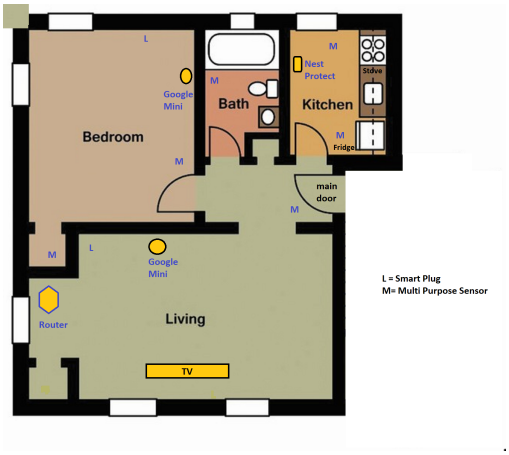


Fig. 1. Sensor map of the Smart Home test environment

The data aggregated is categorized into two types - Behavioral data and Network data. Behavioral data refers to the status of sensors like Sensor ON/OFF and sensor readings. This data is fetched using various API calls from multiple cloud data repositories. On the other hand, Network data refers to the TCP/UDP packet data from IoT devices. We integrate both these data to learn typical behaviors in a smart home environment.

A. Network Data Collection

A large amount of data flows through the gateway router of a smart home. They include data from streaming audio and video. However, we are only interested in the data exchanged from the smart home devices. A packet sniffer monitoring the router alone cannot separate such data from different smart home devices inside the network, as the internal device IP's are masked during Network Address Translation (NAT) in a gateway router. Hence, we altered the normal setup by adding additional devices. The new setup consists of an ARRIS

Sensor	Observations	Sensor	Observations
Bedroom Google Mini	BG_On, BG_Off	Closet	C_Open, C_Close
Bedroom Light	BL_On, BL_Off	Fridge	F_Open, F_Close
Living room Google mini	LG_On, LG_Off	Stove	S_On , S_Off
Living room Light	L_On, L_Off	Phone	P_On, P_Off
Living room door	L_Open, L_Close	Restroom	R_Open, R_Close
Bedroom door	B_Open, B_Close	User	In , Out
Phone	P_On, P_Off	Nest Protect	Ok , N_Ok

TABLE I
LIST OF OBSERVATIONS PER EACH SENSOR

TM822A modem, a NETGEAR R6300v2 wireless router, TP-Link AC1750 dual wireless router, NETGEAR ProSAFE Plus GS105Ev2 switch, and a data collection machine. We used the switch as a bridge between the gateway router and the internal wireless router and enabled port mirroring in the switch. Port mirroring, also known as Switched Port Analyzer (SPAN)[6] is a method of monitoring network traffic by setting up one or more source ports to send a copy of every packet received to a designated destination port. The data collection machine is connected to the destination port of the switch and all data coming from the router will be available in it. Since we manually assigned static IP's to all of the smart home devices and disabled DHCP not to disturb the IP addresses range after certain lease time, we can now effectively filter the required traffic in the data collection machine. Even after filtering data from just the smart home devices, we had a total of 1.1 GB of network data for 3 weeks.

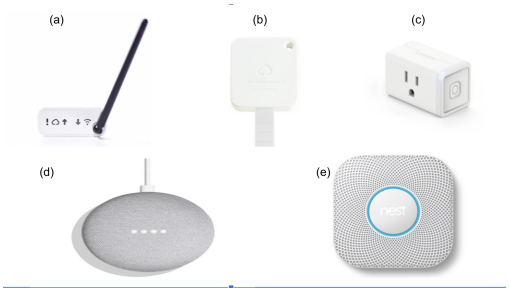
User presence is an important feature used for the identification of anomalous activity. For example, consider a situation in which the main door sensor is opened and the corresponding sensor is activated. If the user is not detected inside the home at this instant, the probability of such an event being an anomaly is high. In our data collection, user's presence in the house is established using the availability of user's smart-phone in the home network. The phone automatically disconnects from the home Wifi network when the user leaves the house and connects to the network when the user is back home. In the data-set, the user's presence is represented as *In* and *Out* events.

Smart plugs(Figure 2 (c)) lets you turn on and off any appliance that plugs into a standard wall socket. In this test environment, the smart plugs are used to control the electric lights and smart plugs can be controlled from phone or Google mini voice assistant. Our smart home setup has two voice assistants, one in the living-Room and the other in the bedroom. Network data from the voice assistants and the smart-phone is collected and is later co-related with smart-plug's behavioral data to determine their mode of operation. In the data-set, the modes of operation are represented as *LG_On, BG_On* and *P_On* events.

[6]https://www.miarec.com/faq/what-is-port-mirroring

B. Behavioral Data Collection

The behavioral data represents the status of doors, stove, fridge, and other devices in the smart home. Sensor tags are used to collect their status. The sensor tags(Figure 2 (b)) monitor the activity of the devices by recording changes in the temperature/humidity/lux/motion. For example, the lux and motion information from a sensor tag connected to the door can be used to detect if it is open or closed. The readings from sensor-tags and smart-plugs are retrieved from the cloud using corresponding API calls of each sensor and it is processed to generate the current status of different smart home components.



(a) Sensor Tag Manager (b) Sensor-tag (c) Smart-plug (d) Google-mini (e) Nest Protect

Fig. 2. List of Sensors deployed in our smart home.

In the living-room, a sensor tag is placed on the main-door to monitor the opening and closing of the door. These events are represented as *L_Open* and *L_Close*. Similarly, a smart-plug is deployed in the living room to control the lighting. The states of the light bulb are updated as *L_On* and *L_Off* events in the data set.

Similarly, in the kitchen, ambient light readings inside the fridge are collected using another sensor-tag. The variation in the degree of brightness inside the fridge is represented as *(F_Open)* and *(F_Close)* events. For the stove, a sensor tag is placed to determine the temperature readings and based on the readings the states of the stove are represented as (*S_On*) and (*S_Off*). A Nest Protect is placed on the kitchen wall to monitor the smoke levels around the kitchen and its alarm status is fetched using Nest API calls.

In the bedroom, two sensor-tags are placed, one on the bedroom door and the other on the closet door. The doors open(*B_Open*, *C_Open*) and close(*B_close*, *C_Close*) events are collected similar to the living room door events. Likewise, a smart-plug is installed in the bedroom to control the bedroom light and its events are stored as (*B_On*) and (*B_Off*).

Each sensor tag uploads its state every 30 seconds while from smart plugs and Nest we fetch status every 30 seconds. By the end of three weeks, the data from all sensors including network data and behavioral data is integrated and sorted based on the time-stamp. Table II shows the sample log of the events generated along with the time-stamp. This data is further pre-processed and modeled according to HMM parameters which are described in detail in sections IV and V.

Time	Sensor	Action	Google-mini	Phone
11:05	User	In		
11:06	Living room door	L_Open		
11:06	Living room door	L_Close		
11:07	Living room Light	L_ON	LG_On	
11:45	Fridge	F_Open		
11:46	Fridge	F_Close		
13:05	Bedroom Door	B_Open		
13:05	Bedroom Light	BL_ON		P_On

TABLE II
SAMPLE LOG COLLECTED FROM THE SENSORS IN OUR SMART HOME

IV. MODELING SMART HOME USING HIDDEN MARKOV MODELS

In this research, we use Hidden Markov Models (HMM) to learn the common behaviors in a smart home. A complete specification of HMM requires us to define parameters *N*, *M*, *A*, *B*, and π [18]. For our smart home setup, we describe each of them below.

N represents the number of states in the model and the set S denotes the set of all states. In a smart home, we model it such that states corresponds to each individual sensor used. In our data-set, the different sensors used are Main-door(MD), Bedroom-door(BD), Closet(C), Fridge(F), Stove(S), Living-room Google-Mini(LG), Light(L), Phone(P), Bedroom Google-mini(BG), Bedroom-light(BL), Nest Protect(N), and User(U). *M* is the number of unique observations possible for each state in *S*. The set V denotes the set of all possible observations in it. In this model, the observation symbols correspond to the physical status of the sensors present in the smart home. Table I illustrates the list of all possible observations from all the sensors. *A* is a matrix which represents the state transition probability distribution as described in equation 1. In a smart home scenario, it represents the sensor transition probabilities. Parameter *B* defines the observation symbol probability distribution and is defined in equation 2. π, which is the initial state probability vector is initialized randomly in our model.

$$
\begin{aligned}
&A = [a_{i,j}] \\
&a_{i,j} = P(q_{t+1} = S_j | q_t = S_i), \\
&\quad where, \\
&\qquad q_t \text{ is the state at time } t, \\
&\qquad S_i \in S(\text{set of all possible states}) \\
&\qquad 1 \leq i, j \leq N, \\
&\qquad t = 1, 2...
\end{aligned}
\tag{1}
$$

$$
\begin{aligned}
&B = [b_{i,j}] \\
&b_{i,j} = P(V_k \text{ at } t | q_t = S_i), \\
&\quad where, \qquad q_t \text{ is the state at time } t, \\
&\qquad v_k \in V(\text{set of all observations}) \\
&\qquad S_i \in S(\text{set of all possible states}) \\
&\qquad 1 \leq i, j \leq N, \\
&\qquad t = 1, 2...
\end{aligned}
\tag{2}
$$

V. LEARNING BEHAVIORS IN A SMART HOME USING HMM

In this section, we describe how we use HMM's to learn the typical behaviors in a smart home. One of the basic problems solved using HMM's is to adjust the model parameters $\lambda = (A, B, \pi)$ to maximize the probability of observation sequences, given the parameters of its model [18]. We use the Baum-Welch or the EM (Expectation Modification) algorithm [18] for this purpose.

We pre-processed each event into a vector of 14 observations. Each observation in the vector represents the active status of the sensors present in the smart home. The combinations of 14 observations together represent an action in the house. Table III presents a sample sequence of vectors extracted from our data-set. For every event generated in the house, the vector is modified with the new status of the sensor that activated the event. The remaining observation values other than the activated sensor remains unchanged. For example, the first row in Table III represents the action "User Enters the House" and the observation of state User is changed from *Out* to *IN*. Similarly, for the second action *'Living-room door Open'*, the observation of Living-room door is changed from *L_Close* to *L_Open* while maintaining the other observations unaltered.

Two sensors Google-mini(LG or BG) and phone(P) are always tagged with each of the lights present in living-room and bedroom. The lights are connected to smart-plugs and can be controlled by either Google-mini or phone. If a light is controlled by Google-mini, the values of states LG, L, and P will be *LG_On, L_On, and P_Off* respectively. Likewise, if the light is controlled by a phone App, the observations for LG, L, and P will be *LG_Off, L_On, and P_On*. For example, row 4 in the Table III is the vector representation of the action *'Living-room Light On by Google-mini'* and so the values of LG, L, and P are *LG_On, L_On, and P_Off*. In a day, if there are 100 events generated, 100 different vectors of 14 observations are created representing the values of the sensors and actions performed. Over a period of three weeks, we extracted 780 sequence of observations from the collected data representing the normal behavior of the household activities in the test smart home environment. For testing, we generate different sequences of observations in a similar way as described above. A detailed description of testing sequences is given in Section VI-B.

After generating these vectors, we used the Baum-Welch algorithm to determine the HMM model parameters $\lambda = (A, B, \pi)$. Now to identify if the new sequences fit to the learned model, we find the log probability of observation sequences using forward-backward algorithm [18]. This probability denotes if the new sequence confirms with the learned model. The probabilities generated are log probabilities and a very high negative value implies very low probability and a small negative value implies high probability. For simplicity, we take the absolute value of this log probabilities in this paper and we name it as the Absolute Log Probability score (ALP_{score}). Hence a very low value for ALP_{score} implies a

high level of conformity to the generated model and a high value for it implies deviation from normal.

VI. Results and Discussion

In this section, we discuss the evaluation of our HMM model using the data collected in section III. In our evaluation, we first determine a threshold and use a k-fold cross validation to identify the efficacy of our approach in different settings. In the next experiment, we generated various abnormal scenarios in a smart home environment to evaluate our model's applicability in identifying anomalous situations. Our experiments are detailed in the following sections.

A. Evaluation under general conditions

Our data-set contains 780 vector sequences. For our first experiment, we split the data-set into two parts. The first 70% of the data is used for estimating the threshold of acceptance to be normal and the rest of it is used for validation. We first trained the HMM model with the first 70% of data and observed that the ALP_{score} ranged between 1 and 7. Hence, we set the threshold value to be 7. This means that any sequence with a value greater than 7 is very unlikely to happen given our observations, and those with values below it are considered normal for the smart home. To validate this threshold, we found the ALP_{score} for the remaining 30% sequences using the already trained model. We found that for 96.8% of the data, we got the ALP_{score} below the threshold of 7. Figure-3 illustrates the plot created for the test-set's accuracy by varying the threshold from 0 to 20.

In our next experiment, we performed a modified k-fold cross validation to determine the efficacy of our approach in a general setting. The value for k is chosen such that each train and test group of data samples is large enough to be statistically representative of the broader data-set.[7]. We choose $k = 6$ and divided the training data-set of 780 sequences into 6 sets. In the first fold, a model is trained using the first 200 sequences of data and is tested using the next 100 sequences. Similarly, in the second fold, a new model is trained using sequences from 200 to 400 and is tested with the next 100 sequences (400 to 500). This process is then repeated for all the 6 folds. The accuracy for the six folds are 98.0%, 99.0%, 99.0%, 95.0%, 99.0%, and 98.0% respectively with an average of 98% test accuracy at a threshold of 7.

B. Abnormal Scenario Detection

To evaluate the model's ability to detect abnormalities, we manually generated a few anomalous scenarios that can happen with the sensors present in the current smart home. An example scenario we manually crafted is *'A confidential closet intrusion with the user not present in the home'*. To simulate this scenario, we took a normal sequence of observation and modified the state of 'Closet' to *C_Open* and state of 'User' to *Out*. Another crafted scenario we generated is *'stove On with user not present in the home'*. This scenario could be an attack to cause physical damage, or might be user oversight, but either case can lead to a disaster. To simulate this scenario, the observations of stove is set to *S_On* and user is set to *Out* in a sequence of normal observations.

[7]https://machinelearningmastery.com/k-fold-cross-validation/

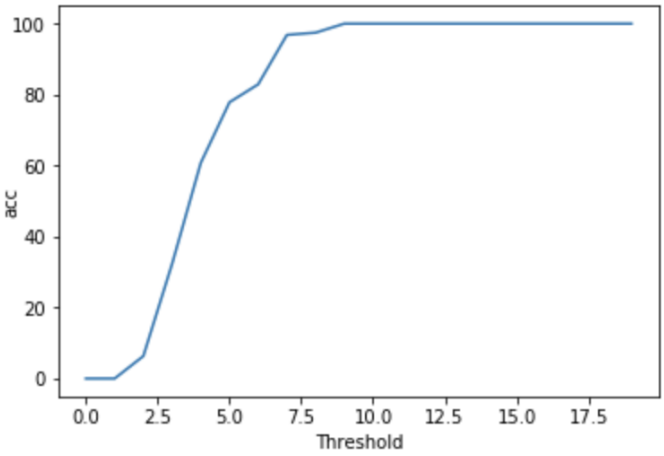


Fig. 3. Plot illustrating rate of accuracy with respect to threshold.

Yet another scenario we developed is using a behavior we observed from the data-set. During the data collection phase, the smart plugs were operated only with Google-mini or phone. They were never controlled physically using the toggle of the smart-plug. In such a case turning on the switch manually will be an anomalous scenario. To test if the model can detect the physical operation of the smart-plugs, we changed the observations of living-room Google-mini, living-room light, and Phone to *LG_Off, L_On, and P_Off*. These observations indicate that the light is turned on without being operated by Google-mini or Phone. Similarly, we crafted five more anomalous scenarios to generate a total 8 anomalous scenarios. For all these crafted scenarios we used the hmm model generated in section VI-A and calculated the ALP_{score} . We got all the values to be above 15 which is well above our threshold and hence we were able to detect all the anomalous scenarios.

In the next experiment, we generated more anomalous sequences by fixing values of specific sensors and then randomly choosing values for other sensors. For example, once we fix the user context as *Out*, any activity related to doors will result in abnormal scenarios. In total, we generated 40 such anomalous scenarios and added them to the 8 manually crafted scenarios creating a combined data-set of 48 anomalous scenarios.

All these scenarios are then tested against the HMM model previously generated in section VI-A. The confusion matrix at threshold 7 for this experiment is presented in Figure 4. We can see that we were able to detect 47 out of 48 attacks with an achieved accuracy of 97%.

VII. Conclusion

In this paper we use a Hidden Markov Model (HMM) for smart home anomaly detection. Various steps followed to setup the test environment, preparation of data-set, and extraction of events from the big data-set have been outlined in detail. We

Row	MD	BD	C	R	F	S	LG	L	P	BG	BL	P	U	N
1	L_Close	B_Close	C_Close	R_Close	F_Close	Stove_Off	LG_On	L_Off	P_Off	BG_On	BG_Off	P_Off	IN	Ok
2	L_Open	B_Close	C_Close	R_Close	F_Close	Stove_Off	LG_On	L_Off	P_Off	BG_On	BG_Off	P_Off	IN	Ok
3	L_Close	B_Close	C_Close	R_Close	F_Close	Stove_Off	LG_On	L_Off	P_Off	BG_On	BG_Off	P_Off	IN	Ok
4	L_Close	B_Close	C_Close	R_Close	F_Close	Stove_Off	LG_On	L_On	P_Off	BG_On	BG_Off	P_Off	IN	Ok
5	L_Close	B_Close	C_Close	R_Close	F_Open	Stove_Off	LG_On	L_On	P_Off	BG_On	BG_Off	P_Off	IN	Ok
6	L_Close	B_Close	C_Close	R_Close	F_Close	Stove_Off	LG_On	L_On	P_Off	BG_On	BG_Off	P_Off	IN	Ok
7	L_Close	B_Open	C_Close	R_Close	F_Close	Stove_Off	LG_On	L_On	P_Off	BG_On	BG_On	P_Off	IN	Ok

MD – Main-door , BD – Bedroom-door, C – Closet, F – Fridge, S – Stove, LG – Living-room Google-Mini, L – Light, P – Phone, BG – Bedroom Google-mini, BL – Bedroom-light, N – Nest Protect, U – User

Row 1 – User Entered Home, Row 2 – Living-room door Open, Row 3 – Living-room door CLOSE, Row 4 – Living-room light turned ON using Google-mini, Row 5 – Fridge Open, Row 6 – Fridge Close, Row 7 – Bedroom door OPEN

TABLE III

A SAMPLE SEQUENCE OF OBSERVATIONS FROM TRAINING DATA

N=100	Actual: YES	Actual: NO	
Predicted: YES	TP = 47	FP = 2	49
Predicted: No	FN = 1	TN = 50	51
	48	52	

Fig. 4. Confusion matrix illustrating prediction rate on test data-set.

have tuned the HMM parameters to maximize the probability of finding the anomalies by learning the typical behaviors in a smart home. An accuracy of 97% is achieved on a set of simulated attack scenarios generated by modifying normal events in the actual data-set. In the future, we plan to add more devices to the test environment and create a more diverse data set. Moreover, in this paper, we assume only a single person as the occupant of the smart home. In our further work, we plan to generalize the system for more occupants. We also need to benchmark the performance of HMM in case of bigger data-sets while also exploring other machine learning and big data techniques to build anomaly detection models for smart homes.

ACKNOWLEDGEMENTS

This research was partially supported by a grant from NIST and a gift from IBM.

REFERENCES

[1] The statistics portal,statistics and studies from more than 22,500 sources, 2017. Last accessed 27 February 2019.

[2] Briana Arrington, LiEsa Barnett, Rahmira Rufus, and Albert Esterline. Behavioral modeling intrusion detection system (bmids) using internet of things (iot) behavior-based anomaly detection via immunity-inspired algorithms. In *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6. IEEE, 2016.

[3] Thi V Duong, Hung Hai Bui, Dinh Q Phung, and Svetha Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-markov model. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 838–845. IEEE, 2005.

[4] Keke Gai, Kim-Kwang Raymond Choo, Meikang Qiu, and Liehuang Zhu. Privacy-preserving content-oriented wireless communication in internet-of-things. *IEEE Internet of Things Journal*, 5(4):3059–3067, 2018.

[5] Keke Gai and Meikang Qiu. Blend arithmetic operations on tensor-based fully homomorphic encryption over real numbers. *IEEE Transactions on Industrial Informatics*, 14(8):3590–3598, 2018.

[6] Keke Gai, Yulu Wu, Liehuang Zhu, Meikang Qiu, and Meng Shen. Privacy-preserving energy trading using consortium blockchain in smart grid. *IEEE Transactions on Industrial Informatics*, 2019.

[7] Karthik Gopalratnam and Diane J Cook. Online sequential prediction via incremental parsing: The active lezi algorithm. *IEEE Intelligent Systems*, 22(1):52–58, 2007.

[8] Edwin O Heierman and Diane J Cook. Improving home automation by discovering regularly occurring device usage patterns. pages 537–540, 2003.

[9] Shrijit S Joshi and Vir V Phoha. Investigating hidden markov models capabilities in anomaly detection. In *Proceedings of the 43rd annual Southeast regional conference-Volume 1*, pages 98–103. ACM, 2005.

[10] Anton Kanev, Aleksandr Nasteka, Catherine Bessonova, Denis Nevmerzhitsky, Aleksei Silaev, Aleksandr Efremov, and Kseniia Nikiforova. Anomaly detection in wireless sensor network of the smart home system. In *2017 20th Conference of Open Innovations Association (FRUCT)*, pages 118–124. IEEE, 2017.

[11] KrebsonSecurity. Mirai botnet authors avoid jail time, 2019. Last accessed 27 Feb 2019.

[12] Sebastian Lühr, Svetha Venkatesh, Geoff West, and Hung H Bui. Explicit state duration hmm for abnormality detection in sequences of human activity. pages 983–984, 2004.

[13] Chuck Martin. North american consumers to have 13 connected devices, 2017. Last accessed 27 Feb 2019.

[14] Trend Micro. New rapidly-growing iot botnet - reaper, 2019. Last accessed 27 Feb 2019.

[15] Nam Thanh Nguyen, Dinh Q Phung, Svetha Venkatesh, and Hung Bui. Learning and detecting activities from movement trajectories using the hierarchical hidden markov model. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 955–960. IEEE, 2005.

[16] Marek Novák, František Jakab, and Luis Lain. Anomaly detection in user daily patterns in smart-home environment. *J. Sel. Areas Health Inform*, 3(6):1–11, 2013.

[17] Meikang Qiu, Zhong Ming, Jiayin Li, Jianning Liu, Gang Quan, and Yongxin Zhu. Informer homed routing fault tolerance mechanism for wireless sensor networks. *Journal of Systems Architecture*, 59(4-5):260–270, 2013.

[18] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[19] Scott Robinson. Smart home attacks are a reality, even as the smart home market soars, 2019. Last accessed 27 February 2019.

[20] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE symposium on security and privacy*, pages 305–316. IEEE, 2010.

[21] Abhinav Srivastava, Amlan Kundu, Shamik Sural, and Arun Majumdar. Credit card fraud detection using hidden markov model. *IEEE Transactions on dependable and secure computing*, 5(1):37–48, 2008.

[22] Yi Zeng, Meikang Qiu, Zhong Ming, and Meiqin Liu. Senior2local: A machine learning based intrusion detection method for vanets. In *International Conference on Smart Computing and Communication*, pages 417–426. Springer, 2018.