# COMPREHENSIVE SMART HOME DEFENSE

Ankit Yadav
Electrical and Computer Engineering
Texas A&M University
College Station, TX 77840 UA
ankityadav270796@tamu.edu

Mahesh Naidu
Electrical and Computer Engineering
Texas A&M University
College Station, TX 77840 USA
mah12man@tamu.edu

## ABSTRACT

There has been a surge in the attacks on home IoT networks and devices that often lack stringent security measures. This motivated us to take up the challenging topic of developing a solution that would protect smart home networks from cyber-offensives. Our research was aimed at analyzing potential threats to a smart home network followed by designing a system that would protect it against such threats. The security system is built using an open-source intrusion detection and prevention system and by adding self- made functionalities to it. It prevents intrusion attempts by unauthorized sources and triggers change of access port in the event of an attack. A change of port further strengthens the security making it difficult to guess the new port. Towards the end, we summarized the work done to achieve our goals and discussed about future scope of work in this research.

## Keywords

Botnet attack, Mirai, Persirai, Deep packet inspection, Attack Signatures

## 1. PRELIMINARIES

*IDS (Intrusion detection system).* An intrusion detection system monitors network traffic through systems to search for suspicious and malicious activity, sending up alerts when it comes across such events.

*IPS (Intrusion prevention system).* An intrusion prevention system also monitors a network for malicious activities however its primary role is to identify intrusion incidents, log corresponding information in a database, report the activity and block further attempts.

*Attack signatures*- are a set of rules or patterns that are used to identify/ classify attacks on a system. A security system would compare these patterns against the contents of queries and responds with an action when a match is observed.

*Deep packet inspection (DPI).* Deep packet inspection (DPI) is a data processing technique or a group of such techniques used for in-depth analysis of packets over a network.

*Whitelists and blaclklists*- IP lists that store IPs identified as malicious are referred to as blacklists and the ones that store trusted IPs are referred to as Whitelists.

## 2. INTRODUCTION

The internet-of-Things describes the ever-growing number of intelligent objects that are being connected to the internet. Almost all home gadgets of today, ranging from an air-conditioner to a speaking toy, have gained many abilities, thanks to the ever-expanding world of the Internet-of-Things. However, as these devices become smarter, they offer the same opportunities of cyber-attacks to cyber criminals as modern computers. IoT centered cyber-attacks hunt for various information, such as, computing and networking resources of a device, the sensitive data it holds and information about other devices connected to it. Once the security of a device is breached, it becomes relatively easier to hack other connected devices and launch large attacks.

In this paper, we first explore the ways home IoT devices can be compromised and how their vulnerabilities could be exploited. Based on this research, we propose a comprehensive solution to prevent intrusion into the home IoT Network.

Home IoT devices have certain vulnerabilities that expose them to cyber- attacks. Currently many devices come with default factory-set usernames and passwords that are weak and can be easily guessed. This effectively weakens the ability of passwords to prevent unauthorized access. The Mirai botnet attack makes use of such vulnerabilities. Mirai identifies vulnerable IoT devices using a table of more than 60 common factory default usernames and passwords, and logs into them to infect them with the Mirai malware. After successful login, it places a bot in these devices which in turn repeat this process: scanning for other devices and compromising those with weak passwords. According to a report released by F5 labs, IoT attacks grew 280% in the first half of 2017, with a large chunk of this growth stemming from Mirai malware [12]. Secondly, many IoT devices are connected to the manufacturer operated cloud services and therefore can be accessed outside the home network. To allow device access, the device must open ports leaving the home network susceptible to cyber- attacks. Another malware called Persirai attacks the universal plug and play protocol to gain access of the IP camera by opening port 81 on the router. Through this open port, the malware injects malicious commands into the camera which then infects other devices that easily connect with it over plug and play.

In this project, we devised techniques that will collectively solve the above problems ensuring security to home networks. Keeping in mind the above problems, we designed an intrusion detection system topped by an intrusion protection system along with a firewall. The intrusion detection system alerts the user about an intrusion. The intrusion protection system blocks a probe from a malicious source. The firewall would prevent a list of IPs identified as malicious from being able to query/ login in the future.

These solutions form the different layers of protection of the total solution.

Research groups in Industry and academia have worked on these problems and have offered various solutions. Our intrusion detection and prevention system will be hosted on the home router which will be the single gateway for the exchange of packets and information between the home IoT devices and the Internet. Since the home router is the focal point of device connectivity management, it will be easier to monitor potential intrusion attempts and prevent attacks at the router level itself. Our solution is built on SNORT which is an open source network intrusion and prevention system, capable of performing real-time traffic analysis and packet-logging on IP networks. Besides, it is can be used to detect probes and attacks from malicious sources and prevent them from making such attempts in the future. Snort was developed by Sourcefire Inc. and is now owned and maintained by Cisco systems.

We have focused our research on certain intrusion methods of botnet attacks which are- 1) ICMP ping floods, 2) suspicious SSH logins attempts and 3) suspicious telnet login attempts. We have devised techniques and algorithms to solve each of these problems. Each of these problems and their solutions will be discussed in the following sections.

## 3. RELATED WORK

Many successful attempts have been made in the past to safeguard IoT devices from intruders and botnet type attacks. Existing works such as Pot2DPI [1] use Honeypot security mechanism together with deep packet inspection to secure Smart home networks. As its name suggests, honeypot is a trap set up to lure malicious attackers towards a port and record their information. Pot2DPI uses a chain of Honeypot and DPI to access, detect and filter suspicious IPs. Honeypot only collects malicious packets on a designated port. Our solution can differentiate between genuine probes and malicious probes on any port based on the attack signatures. There has been significant work done on botnet detection for e.g. BotMiner [10] correlates packets coming from a large number of bots involved in an attack since bots communicate in a similar way and have common malicious activity patterns. In the future, we look forward to utilizing the concepts discussed in [10] and come up with techniques that can fully block botnet attacks having different patterns emanating from the same bots.

## 4. THREAT MODEL

Mirai botnet code infects poorly protected internet devices by logging into those that are still using their factory default username and passwords. The compromised devices are used to launch similar attacks on other devices that have established functional network connections with these devices. We subjected our system to multiple SSH and TELNET login attempts, thus drawing a parallel with Mirai malware. In order to protect the system from intruders, we have set a threshold on the number of failed login attempts by them. If the number of such attempts increase beyond the threshold value, the intruder will be blocked. Another type of attack we have focused on is Ping flood. ICMP ping flood is a simple denial of service attack where the attacker overwhelms the victim with ICMP "echo request" or ping packets. It is referred to as flood because the attacker sends ICMP packets as fast as possible without waiting for any replies effectively leaving the victim system dysfunctional. To prevent against such an attack, we have set a threshold on the number of ICMP pings beyond which the queries will be regarded as malicious and any further probes from the attacker, be it an ICMP ping or an SSH login shall be blocked. We have also implemented the port-rotation feature which will be triggered when a suspicious probe on a port is identified.

## 5. SOLUTION FRAMEWORK

The Snort system has a configuration file called *snort.conf* that it uses at start- up. It defines various settings and configurations necessary to run the program, for e.g., path to different files that will be accessed during program run. A sample snort.conf file is already included in the snort distribution. It can be modified as per the user's requirements. The Snort system can be configured and utilized in three main modes: sniffer, packet logger, and network intrusion detection. In sniffer mode, the program normally will read network packets and display them on the console. It also allows you to save these logs in different files/ format like .csv or SQL database. In packet logger mode, the program will log packets to the disk. In intrusion detection mode, the program will monitor network traffic and analyze it against a rule set defined by the user. The program will then perform a specific action based on what has been identified. The intrusion detection mode of Snort forms the basis of our prevention and detection system. Snort, in its development environment, allows you to write 'rules' which are basically pieces of text/code that evaluate a probe or a query and generate 'alerts' if a certain condition is met. Besides, snort rules also allow blocking of probes from malicious users if the user is 'blacklisted' by the system.

Based on our research of various attacks, we have written rules for the following cases: A) Detect & block ping flood: If the number of ICMP pings exceed a certain number say 100, the source IP will be classified as 'malicious' and any further probes from that IP will be blocked. B) Detect & block unauthorized SSH/ telnet login: If number of failed login attempts exceed a certain threshold say 5, the login attempt will be regarded as an attack and any further probes from that IP will be blocked. C) If a suspicious probe is detected on a port, the system will automatically change the concerned access port and the user will be notified about the attack and the new port.

## 6. METHODOLOGY

When a malicious probe is detected, the corresponding console log is saved to a .csv file. Although the packets are saved to the disk, the console log of a query gives all the information about the packet for e.g. time stamp, type of probe- SSH login or Telnet, source IP address etc. The malicious IP addresses are read from the .csv file mentioned above, a process called signature creation, and then logged

into a snort system file called 'blacklist.rules'. When the program is run, this file is parsed by another rule that blocks 'blacklisted IPs' contained in it from querying the system. However, there is no provision in Snort that dynamically reads the .csv file and adds the intruder's IP address in the blacklist.rules file.

To accomplish this task, we created a 'deep packet inspection' program scripted in python. It will iteratively parse the .csv file for red-flagged packets and send the attacker's IP address to the 'blacklist.rules' file. This signature creation activity takes place dynamically as the 'deep packet inspection' program shall run along with the Snort program.

Figure 1 describes the process map of the intrusion detection and prevention process. All the traffic coming to the Smart home network must pass through the Signature based filter which is built on top of the snort system. All the red-flagged packets are stored in the persistent memory which is later used for deep packet inspection and signature creation. The created signatures are then pushed back to the snort system (IP blacklisting).

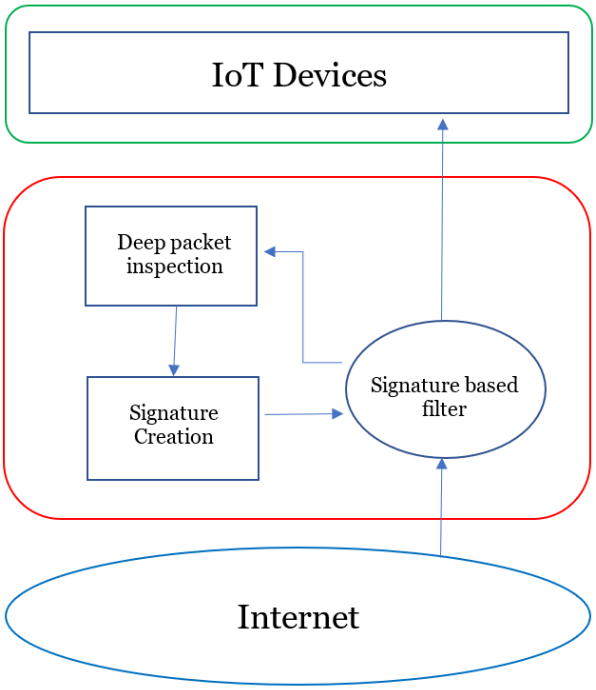


Figure 1: Process map of the smart home intrusion detection and prevention system built on Snort

PORT ROTATION: Currently there has been a rise in the attacks on the services (SSH, TELNET etc.) running on their default ports with weak or default passwords. Mirai and Persirai botnet attacks are very recent examples of such attacks. It has been observed that almost all systems run their SSH and TELNET services on default ports 22 & 23 respectively and therefore fall prey to attackers trying to break into these systems. Configuring different port numbers to access these services considerably reduces the probability of successful attacks by malware. Running these services on different ports would make it difficult for attackers to accurately guess these port numbers that can run into thousands. We have adopted this protection measure in our solution by dynamically allocating a new port to the service every time there is an attack. We have named this feature – Port rotation. As mentioned earlier, many home IoT devices and smart home platforms, e.g. Samsung SmartThings [13] come with cloud-based services for remote access and better management of these devices. To connect and exchange data with their manufacturer operated cloud services, these devices must open their ports or the ports on the router. These ports could be known to an attacker who could gain control of these devices. In the event of an attack, the intrusion protection and detection system would implement the port- rotation functionality described above and notify all parties- the owners, users and cloud- based services about the change of port and the intrusion attempt. We have showcased this attribute by notifying the user about the new-port and also the attack via- email. The overall process of port- forwarding has been described in the diagram in Figure 2.

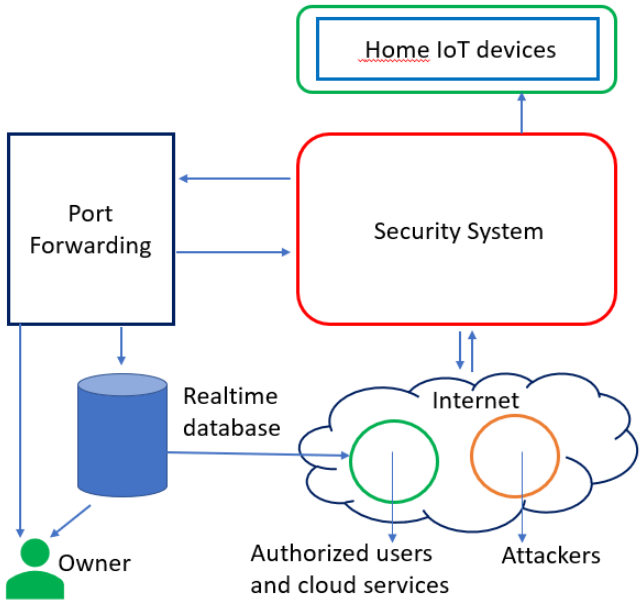


Figure 2: Block diagram describing the mechanism of port-rotation

Figure 3 contains screenshots of the project demonstration. The snort program is running on the first terminal. The deep packet inspection program (python script) is running on the second terminal. The second image shows unauthorized SSH login into the system where the intruder makes 3 failed login attempts. After the third attempt, the intruder is blocked and the SSH port is changed which is shown in the third image. The user is also notified about the change of port via email which is shown in the last image.

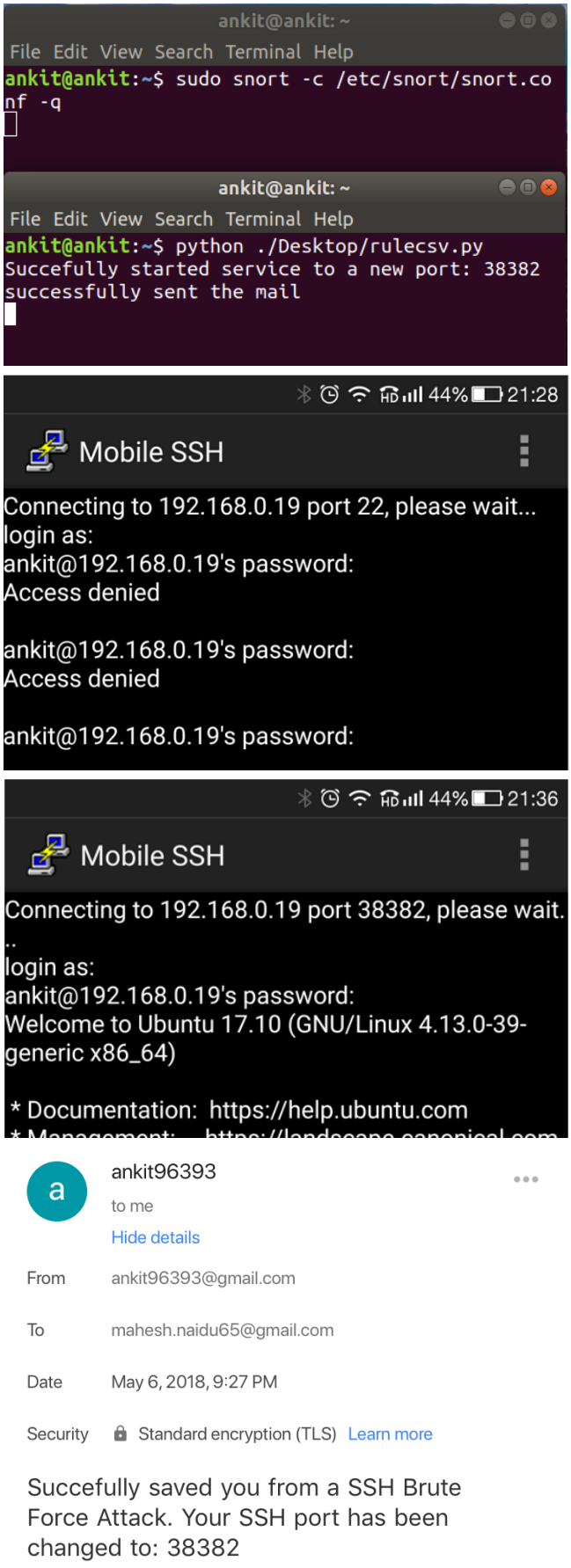


Figure 3: Screenshots of project demonstration

## 7. CHALLENGES

While implementing the details described above, we came across several challenges because of our lack of experience with these types of research problems and shortage of relevant study material. The biggest challenge was selecting the Intrusion Detection and Prevention system. After doing a lot of research, we planned to go ahead with Snort, which is a widely used IDS/IPS system. Still we had little hard time with it because of its poor and not timely updated online documentation. We spent days setting up database-based logging for the snort but only to realize that the support for the database logging has been dropped.

## 8. CONCLUSION & FUTURE WORK

In the earlier part of this report we described what we want to achieve through this project work. We wanted a system which could detect and drop the packets from a malicious or unwanted user depending on the defined fingerprinting rules in snort, but we also wanted to ensure that the authentic users are given uninterrupted access. We have been able to do this with the help of the snort-based filtering and port rotation.

After the incoming packets are found to lie in one of the defined categories, these packets are not delivered to the end devices, instead they are passed on to the Deep Packet inspection module which extracts the important information, e.g. underlying trend in the packets. The extracted information is then used for the creation of new rules dynamically. We have verified the whole idea on services like SSH & TELNET.

Our current work was totally focused on Mirai and Perserai, which is a modified Mirai. Since, the source code of the Mirai has gone open, there has been a surge in the very similar attacks. The solution we defined through this work focuses on the 2 basic attacks: Mirai and Perserai attack. In the future, we would like to come up with a system which can be more general and could prevent our smart homes from the variety of similar attacks.

## 9. ACKNOWLEDGMENTS

We would like to thank Dr. Reyhaneh Safavi Naeini and all the students in the course IoT security and privacy for providing us constant support and suggestions throughout the semester.

## 10. REFERENCES

[1] Martin, Vincentius, Qiang Cao, and Theophilus Benson. "Fending off IoT-hunting attacks at home networks." In Proceedings of the 2nd Workshop on Cloud-Assisted Networking, pp. 67-72. ACM, 2017.

[2] Snort Intrusion Detection and Intrusion Prevention System https://www.snort.org/

[3] Snort User Manuel http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node1.html

[4] Angela D. Orebaugh, Simon Biles, Jacob Babbin. Snort Cookbook. O'Reilly Media, Inc. ©2005

[5] The Reputation Preprocessor in Snort – Blacklists and Whitelists https://sublimerobots.com/2015/12/the-snort-reputation-preprocessor/

[6] Brandon Rice. Automated snort signature generation http://commons.lib.jmu.edu/cgi/viewcontent.cgi?article=1314&context=master201019

[7] David Elson. Intrusion Detection, Theory and Practice https://www.symantec.com/connect/articles/intrusion-detection-theory-and-practice

[8] Mirai Source Code (For Educational Purposes only) https://github.com/Screamfox/-Mirai-Iot-BotNet/blob/master/TUTORIAL.txt

[9] http://books.gigatux.nl/mirror/snortids/0596006616/snortids-CHP-7-SECT-3.html

[10] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. 200. BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-independent Botnet Detection. In *Proceedings of the 17th Conference on Security Symposium*.

[11] https://www.techrepublic.com/article/report-iot-attacks-exploded-by-280-in-the-first-half-of-2017/

[12] http://docs.smartthings.com/en/latest/architecture/#smartthings-cloud

[13] Steve Ragan. Mirai ID/Password Collection https://www.csoonline.com/article/3126924/security/here-are-the-61-passwords-that-powered-the-mirai-iot-botnet.html