**IOT Botnet Attack detection**

The Internet of Things (IoT) describes a world where just about anything is an Internet-enabled device. IoT is comprised of smart physical objects such as vehicles and buildings or embedded devices such as refrigerators, toasters and routers. These devices feature sensors and an IP address for Internet connectivity, enabling these objects to collect and exchange data while allowing users the ability to automate or control their devices.

Currently, more things are connected to the Internet than people. According to Gartner, there are approximately 6.4 billion connected devices in use worldwide (2016), estimated to reach 20 billion by 2020.1 A staggering number that exponentially expands a hacker’s field of attack.

**1. The Lack of Security in IOT**

The existence of vulnerabilities is natural in everything, but after 48 years since the first network link being established, developers must be aware of possible vulnerabilities and must take extra care in designing these networks to avoid the being-under-attack phenomenon. The authors of “Botnets and internet of things security” has mentioned the most common IoT vulnerabilities. A news report published by Oxford Press has also addressed these same vulnerabilities. The most relevant vulnerabilities to DDoS have been listed in Table 1 along with theoretical solutions. The most common attacks or problems that arise due to the vulnerabilities in Table 1 are addressed as; 1) Using unauthorised access to control other users access rights, 2) Unused ports can be back doors to attackers to carry out malicious acts, 3) Data is sent through insecure network with no encryption can be compromised easily, and 4) Software updates contain information that should not be public and is not hidden. At least 70% of today’s IoT devices hold such risks (but with no advice on how to prevent them). Vulnerabilities are one of the reasons botnets, such as Mirai or Hajime botnets, are able to get control of IoT devices and use these devices to flood servers leading to DDoS. These botnet malwares are a result of open ports and unchanged default passwords.

The research conducted by authors of the above paper are used as a foundation for further complex attacks and botnets which will explained moreover. The author of “Smart Real-Time Internet-of-Things Network Monitoring System” has introduced three more security controls including secure booting; checking that there are no malicious acts undertaking while the system is running. The second is the obligation of access control at an Operating System level. The last one is utilising firewalls on the IoT devices. These controls are designed to make IoT manufacturers more conscious of what must be included in their design before any implementation takes place. This makes sense, a user does not want to add a burden on them while using devices that are meant to add simplicity to their lifestyle. Businesses must provide reliable and secure products to retain their reputation.

|  |  |
| --- | --- |
| **Vulnerability** | **Solution** |
| Insufficient authentication and authorisation | Changing default passwords and access rights |
| Insecure network services | Monitoring ports and any malicious activity |
| Lack of transport encryption and integrity verification | Encrypt all files and data |
| Insecure software or firmware | Along with encryption, software must be regularly checked for bug fixes and new security bugs |

One good step in the prevention of DDoS and DoS is done by Radware. They have produced a handbook that explains nearly everything that is related to DDoS attacks including HTTP flooding. It even explains that even SSL and HTTPS can be flooded by requests which shows that encryption is not a great solution in preventing DDoS at all. This book is highly recommended for anyone who wants to know more about how DDoS can happen and what security measures must be done giving good guidance on what to consider. Vulnerabilities are not the only cause of DDoS or DoS, the authors of “Security attacks in IoT: A survey” have mentioned important artefacts about including various attacks leading to DoS. One factor is physical security, this refers to physical damage. A broken device or server will obviously not work, causing DoS. Other causes that were mentioned include node jamming and radio interference. Their work touches and compares the possibility of DoS from many different angles, but unfortunately does not discuss any solutions to follow. It only gives suggestions about how network technologies can be made into lightweight networks to improve security in the future

## 2. The Botnet Evolution

With the number of IoT devices dramatically accelerating, there is corresponding increase in the number of botnets and cyber-attacks. Let’s take a look at botnets: traditional and IoT.

* Traditional Botnets: A traditional botnet is a collection of compromised computers or servers, often referred to as zombies, infected with malware that allows an attacker to control them, carrying out tasks on their behalf. Botnet owners or herders are able to control these infected machines in the botnet by means of a covert channel such as Internet Relay Chat (IRC) or peer-to-peer. These control methods issue commands to perform malicious activities such as distributed-denial-of-service (DDoS) attacks, spam mail or information theft.
* IoT Botnets: An IoT botnet is a collection of compromised IoT devices, such as cameras, routers, DVRs, wearables and other embedded technologies, infected with malware. This malware allows an attacker to control the devices, carrying out tasks just like a traditional botnet. Unlike traditional botnets, infected IoT devices seek to spread their malware, persistently targeting more and more devices. While a traditional botnet may consist of thousands or tens of thousands of devices, an IoT botnet is larger in scale, with hundreds of thousands of compromised devices.

## 3. The Impact

As IoT botnets continue to grow, they are being leveraged to launch DDoS attacks. Because IoT devices are Linux and Unix-based systems, they often are targets of executable and linkable format (ELF) binaries, a common file format found in embedded systems’ firmware (see Figure 1). The malware delivery method typically targets SSH or Telnet network protocols by exploiting default, hardcoded credentials or simply brute-forcing techniques. Once compromised, the malware payload is delivered to the device for enrollment into the botnet.

Because IoT devices are “always on” (24/7/365) and ready to go, an IoT “bot herder” can build and deploy large- scale attacks within minutes, such as a massive 1Tbps DDoS attack.



Figure 1: Example of format (ELF) binaries

**4. Types of IoT Botnets**

Because IoT includes a vast and ever-growing array of networked devices (e.g., smart meters, medical devices, public safety sensors, etc.), many IoT botnets such as Aidra, Bashlite and Mirai can utilize scanners designed to locate exposed ports and default credentials on these devices (see Figure 2).



Figure 2: Default passwords

Here’s a sample of some large scale IoT botnet attacks:

* Linux.Aidra – Also known as Linux.Lightaidra, this botnet was discovered in 2012 by security researchers at ATMA.ES. It was first noticed when researchers witnessed a large number of Telnet-based attacks on IoT devices.
* Bashlite – Also known as Gayfgt, Qbot, Lizkebab and Torlus, this IoT botnet was discovered in 2014 with the Bashlite source code published (with several variants) in 2015. Some variants of this botnet reached over 100,000 infected devices, serving as the precursor to Mirai (see below).
* Mirai – Gaining worldwide attention in 2016, the Mirai botnet consisted of record-breaking DDoS attacks on Krebs, OVH and Dyn. The botnet, which targeted closed-circuit television cameras, routers and DVRs, generated traffic volumes above 1Tbps. Featuring ten pre-defined attack vectors, this botnet took down the infrastructure of service providers and cloud scrubbers. Some of the vectors include GRE floods and Water Torture attacks.
* Linux/IRCTelnet – Discovered in 2016 by Malware Must Die, this IoT botnet targets routers, DVRs and IP cameras. It can send UDP and TCP floods along with other methods in both Ipv4 and Ipv6 protocols.

## 5. Why IoT Botnets?

There are a number of critical reasons attackers target IoT devices.

* Low-hanging fruit as embedded devices are easily exploited (e.g., default credentials, exposed services)
* Always-on devices with 24/7/365 availability and explosive marketplace growth
* Off-the-shelf products with low security standards (often root:root and admin:admin since few end users change this nomenclature once deployed)
* Malware can easily change default passwords, preventing a user from logging in or other attackers taking control
* Devices are rarely monitored and poorly maintained, allowing hackers to easily shut down or enslave large numbers of IoT devices
* Low cost of entry for attackers as control of thousands of devices can occur for nearly zero cost (i.e., different than the high cost of accessing and controlling servers for more traditional DDoS attacks)

## 6. The Marketplace

Since the publication of IoT botnets such as Bashlite and Mirai, more botnets have emerged, with attackers scanning for new victims to target. Marketplace vendors see financial gain from the work of these hackers, offering IoT botnets for a fee; for example, Mirai for DDoS-as-a-Service. Many notorious DDoS groups have already entered the DDoS-as-a-Service business, monetizing their capabilities by renting out their powerful stresser services. These new, off-the- shelf attack services, for as little as $19.99/month, are commoditizing the art of hacking, making it possible for novice hackers to launch DDoS attacks over 100Gbps.

Vendors of these IoT botnet services are market savvy, utilizing digital marketing such as weekly promotions and Web banners to reach potential buyers (see Figure 3). And they are not stopping with the Clearnet and Darknet, but advertising on social media including Facebook and Twitter (see Figure 4).

## 7. Vigilante Hackers



Figure 4: Social Media Advertisement

Recently, a number of vigilante hackers have attempted to secure or lockout other bot herders from IoT devices via their own malware. One of the original examples of this was a botnet named Linus.Wifatch, but recently two new vigilante botnets have been discovered. In 2017, Radware’s research team monitored thousands of attempts from Hajime, an alleged vigilante botnet, when it discovered BrickerBot.

## Linux.Wifatch

A group called White Team released a piece of malware in 2014 known as Linux.Wifatch. Designed to infect routers to prevent them from being infected by other IoT botnets, Wifatch is a peer-to-peer botnet that stayed updated of evolving threats so it could attempt to mitigate them as well. It infects devices via Telnet by leveraging default credentials, closes the Telnet session and instructs the owner to login and change their Telnet password and update the firmware.

## Hajime

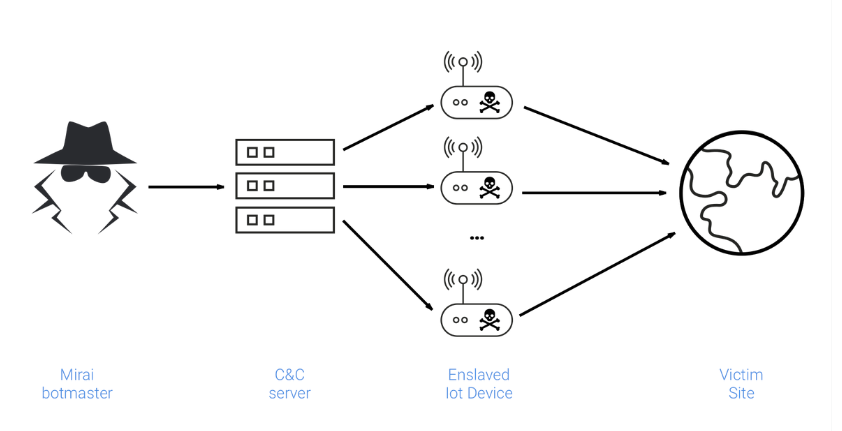
Hajime is a sophisticated, flexible and future-proof IoT botnet. It is capable of updating itself and provides the ability to extend its member bots with ‘richer’ functionality efficiently and fast. The author behind Hajime is another suspected vigilante hacker attempting to secure IoT devices. This is a peer-to-peer botnet that has infected over 300,000 devices to date. Hajime targets devices via Telnet and gains access by brute-forcing default credentials. Hajime was released prior to Mirai but targets devices like routers, DVRs and CCTVs.

## Brickerbot

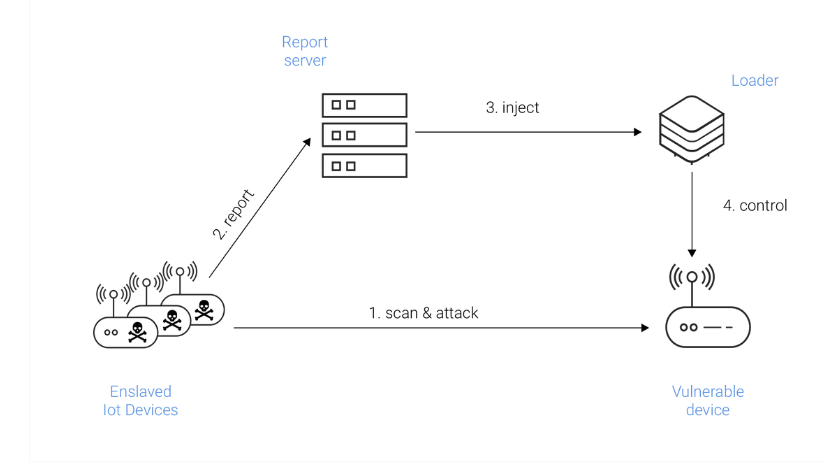
Earlier this year, Radware identified a new botnet named BrickerBot. BrickerBot uses a network of globally- distributed devices that passively detect exploit attempts from devices infected with IoT bots such as Mirai. BrickerBot reacts to an exploit attempt by scanning the source of the exploit for a set number of ports in an attempt to secure the device. If it is unable to, BrickerBot launches a permanent denial-of-service (PDoS) attack that attempts to brick the infected device by leveraging 90-brick sequences via a Telnet session. As long as an IoT device does not become infected by malware, there should be no reason to fear BrickerBot.

**8. How Mirai Works?**

At its core, Mirai is a self-propagating worm, that is, it’s a malicious program that replicates itself by finding, attacking and infecting vulnerable IoT devices. It is also considered a botnet because the infected devices are controlled via a central set of command and control (C&C) servers. These servers tell the infected devices which sites to attack next. Overall, Mirai is made of two key components: a replication module and an attack module

Attack Module

## **8.1 Replication Module**

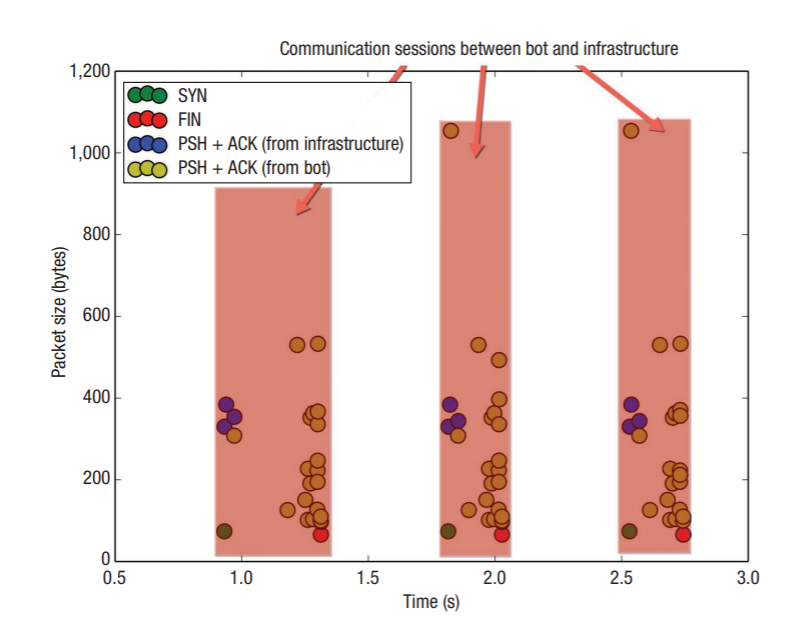
* 1. The replication module is responsible for growing the botnet size by enslaving as many vulnerable IoT devices as possible. It accomplishes this by (randomly) scanning the entire Internet for viable targets and attacking. Once it compromises a vulnerable device, the module reports it to the C&C servers so it can be infected with the latest Mirai payload, as the diagram above illustrates.

To compromise devices, the initial version of Mirai relied exclusively on a fixed set of 64 well-known default login/password combinations commonly used by IoT devices. While this attack was very low tech, it proved extremely effective and led to the compromise of over 600,000 devices. For more information about DDoS techniques, read this Cloudflare primer.

## **8.2 Mirai botnet analysis and detection**

The good folks at Imperva Incapsula have a great analysis of the Mirai botnet code. You should head over there for a deep dive, but here are some of the high points:

* Mirai can launch both HTTP flood and network-level attacks
* There are certain IP address ranges that Mirai is hard-wired to avoid, including those owned by GE, Hewlett-Packard, and the U.S. Department of Defense
* Upon infecting a device, Mirai looks for other malware on that device and wipes it out, in order to claim the gadget as its own
* Mirai's code contains a few Russian-language strings—which, as we later learned, were a red herring about its ultimate origins

Distinctive communication patterns between an infected IoT device and Mirai’s loader component. SYN (synchronize), FIN (finish), PSH (push), and ACK (acknowledge) are standard TCP packet types.

Imperva Incapsula also has a tool that will scan your network looking for vulnerabilities, particularly looking for devices that have the logins and passwords on Mirai's list. Because Mirai stores itself in memory, rebooting the device is enough to purge any potential infection, although infected devices are generally re-infected swiftly. Therefore, the recommendation is to change the password to something stronger before rebooting if you have any vulnerable devices.

**8.3 Mirai Variants**

One would have expected the public release of Mirai’s source code, coupled with its relatively noisy network presence, to quickly lead to effective detection and defense mechanisms. However, the opposite occurred: within only two months of the source code’s release, the number of bot instances more than doubled, from 213,000 to 493,000, and a wide range of Mirai variants emerged.11 Even today—nearly a year after Mirai’s appearance—bots continue to exploit the same weak security configurations in the same types of IoT devices.

Although most Mirai infections occur through TCP ports 23 and 2323, Mirai strains identified in November 2016 rely on other TCP ports to commandeer devices—for example, port 7547, which ISPs use to remotely manage customers’ broadband routers. That same month, one such Mirai variant knocked nearly a million Deutsche Telekom subscribers offline.

In February 2017, a Mirai variant launched a 54-hour-long DDoS attack against a US college. The following month, yet another novel variant appeared with bitcoin miner functionality, although it’s doubtful that compromising IoT devices would yield significant revenue.

Active since April 2017, Persirai is another IoT botnet that shares Mirai’s code base. Discovered by Trend Micro researchers and named for its likely Iranian origin (the name is a portmanteau of Persian and Mirai), it attempts to access the interface of specific vendors’ webcams through TCP port 81. If successful, it then worms its way into the client’s router through a universal plug and play (UPnP) vulnerability, downloads the malicious binaries, and, after execution, deletes them. Rather than deducing webcam credentials via a brute-force attack, the malware proliferates by exploiting a documented zero-day flaw that lets attackers directly obtain the password file. The DDoS attack armory includes User Datagram Protocol flooding attacks. An estimated 120,000 devices in the wild are vulnerable to Persirai.

**8.4 Other IOT Botnets**

Following Mirai’s example, other IoT botnets have recently emerged. While relying on the same basic principles, the authors of this malware are exploring increasingly sophisticated mechanisms to make their botnets more powerful than the competition as well as to obfuscate their activity.

The first IoT botnet written in the Lua programming language was reported by MalwareMustDie in late August 2016. Most of its army is composed of cable modems with ARM CPUs and using Linux. This malware incorporates sophisticated features such as an encrypted C&C communication channel and customized iptables rules to protect infected devices.

The Hajime botnet, discovered in October 2016 by Rapidity Networks, uses a method of infection similar to that of Mirai. However, rather than having a centralized architecture, Hijame relies on fully distributed communications and makes use of the BitTorrent DHT (distributed hash tag) protocol for peer discovery and the uTorrent Transport Protocol for data exchange. Every message is RC4 encrypted and signed using public and private keys. So far, Hajime hasn’t evidenced malicious behavior; in fact, it actually closes potential sources of vulnerabilities in IoT devices that Mirai-like botnets exploit, causing some researchers to speculate that it was created by a whitehat. But its true purpose remains a mystery.

A BusyBox-based IoT botnet like Mirai, BrickerBot was unearthed by Radware researchers in April 2017. By leveraging SSH service default credentials, misconfigurations, or known vulnerabilities, this malware attempts a permanent denial-of-service (PDoS) attack against IoT devices using various methods that include defacing a device’s firmware, erasing all files from its memory, and reconfiguring network parameters.

## **9. Botnet Detection and Removal Tools**

Botnet detection can be difficult, since infected bots are designed to operate without users knowing about them. A blog post from CA Technologies suggests **several symptoms of botnet infection that administrators should look for**. These Include:

* Internet Relay Chat traffic (botnets and bot masters use IRC for communications)
* Connection attempts with known command-and-control servers
* Multiple machines on a network making identical DNS requests
* High outgoing Simple Message Transfer Protocol traffic (as a result of sending spam)
* Unexpected pop-ups (as a result of clickfraud activity)
* Slow computing/high CPU usage spikes in traffic, especially on Port 6667 (used for IRC), Port 25 (used in email spamming) and Port 1080 (used by proxy servers)
* Outbound messages (email, social media, instant messages, etc.) that weren’t sent by the user

Some tools, such as [CDW’s Threat Check tool](https://www.cdw.com/content/cdw/en/solutions/cybersecurity/security-threat-check.html?cm_mmc=Vanity-_-CDWThreatCheck-_-NA-_-NA), perform passive inspection of all inbound and outbound network traffic and look for evidence of malicious activity. “It will not block any traffic but simply monitor and report on what it sees. This includes connections to botnets, connections to command and control servers, remote access tools, visits to sites hosting malicious code, or any other evidence of an infection,” Aaron Colwell, manager of strategic software sales for the analytics practice at CDW, [writes on CDW’s solutions blog](https://blog.cdw.com/security/is-your-network-under-attack).

“Botnet detection is useless without having **botnet removal capabilities**,” the CA blog notes. “Once a bot has been detected on a computer, it should be removed as quickly as possible using security software with botnet removal functionality.”

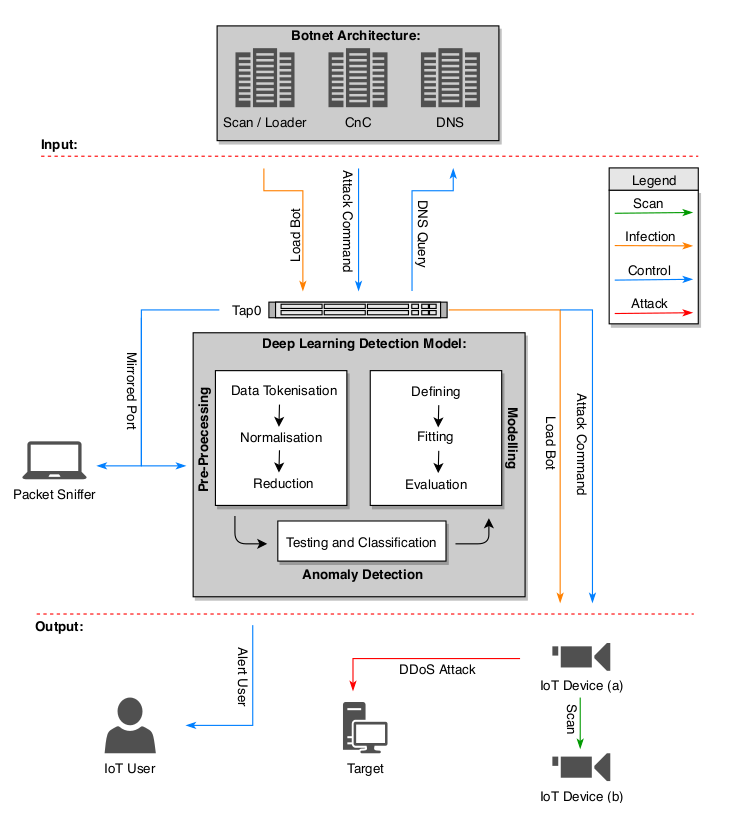
Microsoft offers tools to [remove malicious software](https://support.microsoft.com/en-us/help/890830/remove-specific-prevalent-malware-with-windows-malicious-software-remo), as do many other [security software companies](https://www.cdw.com/search/?wclss=f14&pcurrent=1&cm_re=hubpa-_-category-_-1).

**10. Attack Detection**

**10.1 Deep Learning for Attack Detection**

The increasing presence of IoT systems in a broad range of applications, as well as their increasing computing and processing capabilities make them a valuable attack target, such as network packets and malware designed to compromise specific IoT devices. Attack detections in IoT systems is notably different from the existing mechanisms because of the special service requirements, such as low latency, resource specificity, distributed nature, mobility, to mention a few. This means that conventional network attack detection has limited application in addressing IoT security problems. According to Kaspersky Lab, in 2016 the majority of IoT devices examined were insecure, using default passwords or unpatched vulnerabilities, and easily compromised by Mirai and Hajime malware.A considerable number of zero-day attacks are continuously emerging because of the addition of various IoT protocols. Most of these attacks are small variants of previously known cyber-attacks that present a difficulty in their detection even for advanced computational intelligence mechanisms such as traditional machine learning systems.

Previous literature have suggested the potential of leveraging machine learning to enhance security threat hunting, but it is not practical to simply integrate machine learning in static and dynamic cyber security analysis due to the wide variety and distribution of IoT devices, particularly for (inexpensive) IoT devices with limited processing power. On the other hand, the success of deep learning (DL) in various big data fields has attracted noticeable interest in cybersecurity fields. The application of DL has become practical because of the advances in computer architecture (e.g. NVIDIA DGX platforms) and in development of new neural network libraries (such as Theano and Tensorflow for instance); also, the availability of large and diverse training datasets made a contribution to the effectiveness of deep learning algorithms.



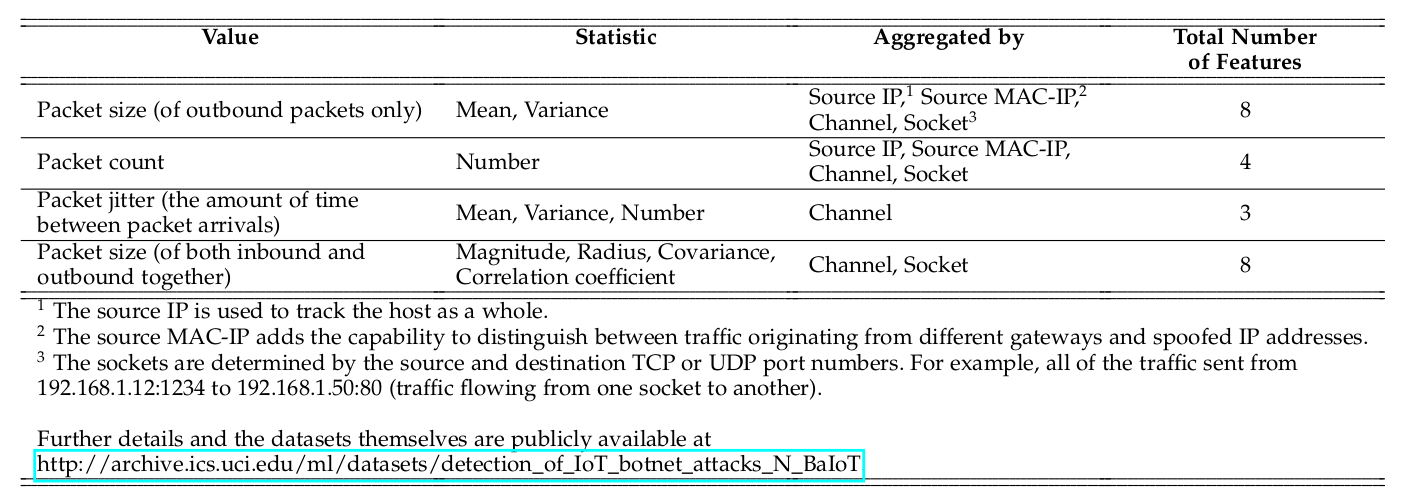
Botnet Architecture and Deep Learning Detection Model

Deep learning (DL) enables several breakthroughs of conventional AI tasks in the fields of image processing, pattern recognition and computer vision. Deep networks are capable of achieving significant improvement in accuracy of classification and predictions in these complex tasks. The main benefit of deep learning is the absence of manual feature engineering, unsupervised pre-training and compression capabilities which enable the application of deep learning feasible even in resource constraint networks. It means that the capability of DL to self-learning results in higher accuracy and faster processing, which can be effectively utilised for a novel distributed attack detection in IoT systems. This is very important in the context of IoT security because such systems face a plethora of security problems, including jamming, spoofing, replaying and eavesdropping, but also prone to issues related to resource constraints e.g. out-of-memory accesses, unsafe programming languages, etc.

This research is aimed at adopting a deep learning approach to cybersecurity to enable the detection of botnet attacks. Other machine learning and evolutionary computing techniques have been successfully applied in mitigating against botnet attacks. One example is the use of swarm intelligence for destroying any rigid master-slave relationship between bots and for autonomizing the bot operating roles. The evolving behaviour of botnets often enables them to circumvent the traditional detection approaches. The development of behavioural detection approaches, however, have helped in dealing with the constant change in the botnet activities by finding the common patterns that botnets follow across their life cycle. For instance, all the bots need connect to the C&C server to receive new orders, and this kind of behaviour observed only after a long period of time can guide the detection methods.

**10.2 Network-based Detection of IoT Botnet Attacks Using Deep Autoencoders**

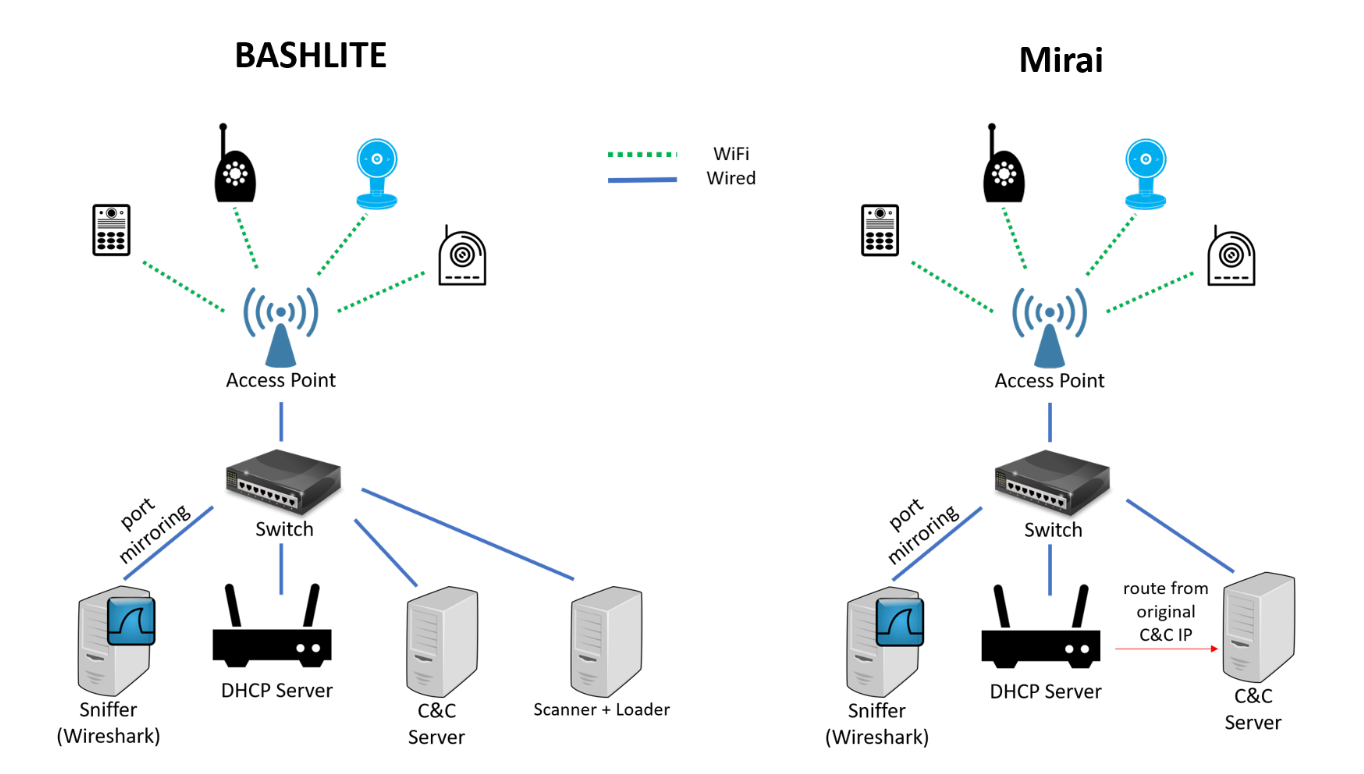
The method proposed for detecting IoT botnet attacks relies on deep autoencoders for each device, trained on statistical features extracted from benign traffic data. When applied to new (possibly infected) data of an IoT device, detected anomalies may indicate that the device is compromised. This method consists of the following main stages: (1) data collection, (2) feature extraction, (3) training an anomaly detector, and (4) continuous monitoring. Data collection. They capture the raw network traffic data (in pcap format) using port mirroring on the switch through which the organizational traffic typically flows. To ensure that the training data is clean of malicious behaviors, the normal traffic of an IoT is collected immediately following its installation in the network. Feature extraction. Whenever a packet arrives, they take a behavioral snapshot of the hosts and protocols that communicated this packet. The snapshot obtains the packet’s context by extracting 115 traffic statistics over several temporal windows to summarize all of the traffic that has (1) originated from the same IP in general, (2) originated from both the same source MAC and the same IP address, (3) been sent between the source and destination IPs (channel), and (4) been sent between the source to destination TCP/UDP sockets (socket).

Table 2

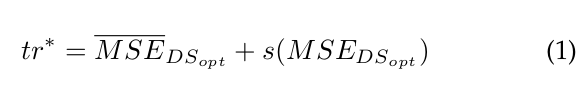
They extract the same set of 23 features (capturing the above, see Table 2) from five time windows of the most recent 100ms, 500ms, 1.5sec, 10sec, and 1min. These features can be computed very fast and incrementally and thus facilitate real time detection of malicious packets. Additionally, although generic these features can capture specific behaviors like source IP spoofing, characteristic of Mirai’s attacks. For instance, when a compromised IoT device spoofs an IP, the features aggregated by the Source MAC-IP, Source IP and Channel will immediately indicate a large anomaly due to the unseen behavior originating from the spoofed IP address.

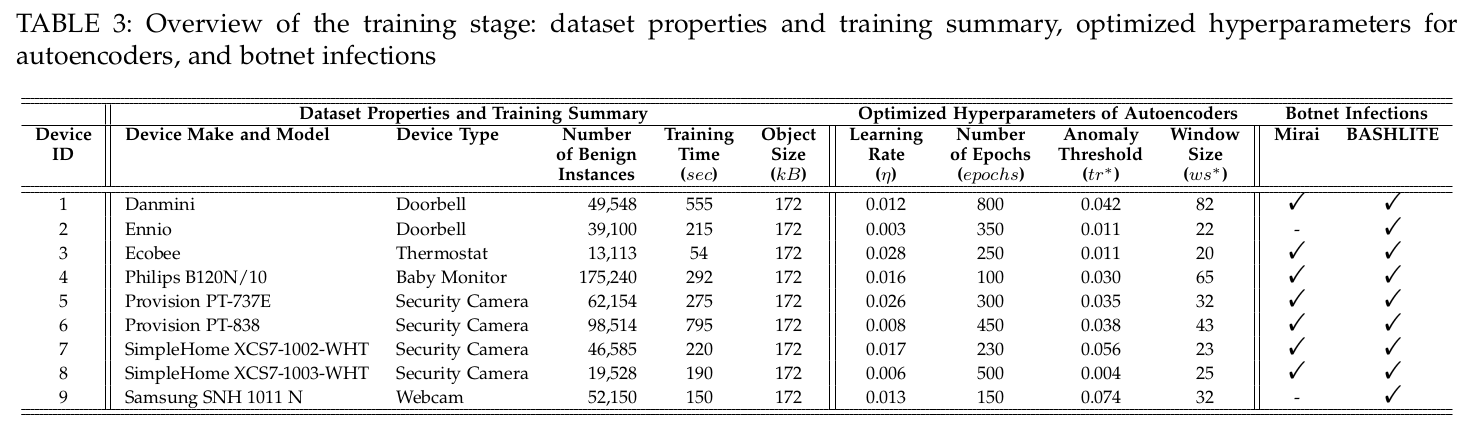
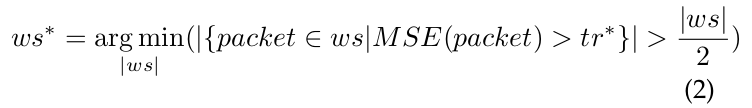
**Training an anomaly detector**. As our base anomaly detector, they use deep autoencoders and maintain a model for each IoT device separately. An autoencoder is a neural network which is trained to reconstruct its inputs after some compression. The compression ensures that the network learns the meaningful concepts and the relation among its input features. If an autoencoder is trained on benign instances only, then it will succeed at reconstructing normal observations, but fail at reconstructing abnormal observations (unknown concepts). When a significant reconstruction error is detected, then they classify the given observations as being an anomaly.

They optimize the parameters and hyperparameters of each trained model such that when applied to unseen traffic the model maximizes the true positive rate (TPR, detecting attacks once they occur) and minimizes the false positive rate (FPR, wrongly marking benign data as malicious). For training and optimization, they use two separate datasets which only contain benign data, from which the model learns patterns of normal activity. The first dataset is the training set (DStrn), and it is used for training the autoencoder, given input parameters such as the learning rate ( η , the size of the gradient descent step), and the number of epochs (complete passes through the entire DStrn). The second dataset is the optimization set (DSopt), and it is used to optimize these two hyperparameters ( η and epochs ) iteratively until the mean square error (MSE) between a model ’s input (the original feature vector) and output (the reconstructed feature vector) stops decreasing. Stopping at this point prevents overfitting DStrn , thus promoting better detection results with future data. DSopt is later used to optimize a threshold (tr) which discriminates between benign and malicious observations; finally, it is also used to optimize the window size ( ws ), by which the FPR is minimized.

Lab setup for detecting IOT botnet attacks

Once the model training and optimization is complete the tr\* is set. This anomaly threshold, above which an instance is considered anomalous, is calculated as the sum of the sample mean and standard deviation of M SE over DS opt (see Equation 1).

Preliminary experiments revealed that deciding whether a device’s packet stream is anomalous or not based on a single instance enables very accurate detection of IoT-based botnet attacks (high TPR). However, benign instances were too often (in approximately 5-7% of cases) falsely marked as anomalous. Thus they base the abnormality decision on a sequence of instances by implementing a majority vote on a moving window. They determine the minimal window size ws\* as the shortest sequence of instances, a majority vote which produces 0% FPR on DSopt (see Equation 2).

**Continuous monitoring for anomaly detection.** Eventually, they apply the optimized model to feature vectors extracted from continuously observed packets, to mark each instance as benign or anomalous. Then, a majority vote on a sequence (the length of ws\* ) of marked instances is used to decide whether the entire respective stream is benign or anomalous. Consequently, an alert can be issued upon the detection of an anomalous stream, as it might indicate malicious activity on the IoT device.