An Analysis of the DDoS Potential of Single Board Computers: A Raspberry Pi Case Study

Martijn Bruning
University of Twente,
The Netherlands
m.s.bruning@student.utwente.nl

ABSTRACT
In early 2012 the Raspberry Pi was released in the UK with the intention of promoting basic computer science in schools. The Pi is a credit card sized computer board that has most of the same functionality as a desktop computer. Since the inception of the Pi there have been numerous other Single Board Computers (SBC), Beaglebone [4] and HummingBoard [17] are competitors with the same concept as the Raspberry Pi. Another competitor, Arduino [1], was even released some years prior. The number of competition is a clear sign that the popularity of SBC is on the rise.

Along with the rise of these computers, Readily Bootable OS Images (RBOI) have been developed specifically for use with such devices. A notable RBOI example is Kodi, a home theater centre oriented OS. Other already existing OS Images have also found their way onto the SBC. Using online instructions a user can have the device up and running in a matter of minutes and with a few extra steps it becomes easy to enable remote access. Without proper attention to security such as setting up a custom login instead of using the default one, devices become more vulnerable, making it easier for attackers to find and infect them. Creating a botnet out of these systems for use in a Distributed Denial of Service (DDoS) attack could become more interesting for attackers as the number of SBC increases over time. Subsequently this paper proposes research into the potency of a botnet consisting of these devices executing a DDoS attack by performing an analysis of network traffic.

Keywords
Arduino, Beaglebone, Denial of Service, HummingBoard, Single Board Computers, Readily Bootable OS Images, Raspberry Pi, Security

1. INTRODUCTION
The number of Out of the Box (OotB) SBC, like the Raspberry Pi, that have come into use have shown a steady increase in the past few years [19]. Devices such as the Raspberry Pi share the same functionality as regular home computers such as running applications and web browsing, and have become widespread in use for a number of purposes. The modularity of the devices and the ability to program on them have made them especially interesting for schools wanting to enhance the education of younger students [18].

While it is possible for a more advanced user to install an image themselves, most often SBC are run with a RBOI that comes preinstalled on an external storage device. There are also some vendors that provide convenient software, such as New Out Of The Box Software (NOOBS) for the Raspberry Pi, that make it easy to install OS images onto a storage device such as an SD card. Any of the options make it exceedingly easy for users to start working with these devices without needing much technical knowledge. Without properly instructing users about SSH security it becomes likely that some users start working with the device in an insecure way. A number of bootable OS images come with SSH enabled by default [16]. For the images that do not feature default SSH it is exceedingly easy to find a step by step guide that instructs you how to set it up [15]. Connecting these devices to the Internet before changing the default login and password makes it a simple task for a malicious user to take over the devices and infect it with malware.

With their growing sophistication, botnets form a threat as powerful as that of viruses, worms and other familiar cyber threats [2]. With more and more systems being connected to the Internet a powerful DDoS attack can cause serious financial damage [3] or worse, destabilise Critical Information Infrastructure (CII) such as Supervisory Control And Data Acquisition (SCADA) systems that are responsible for managing power plants and floodgates [22]. In 2010 Stuxnet has shown us just how vulnerable these systems really are [11]. Consequently the increasing number of OotB devices such as the Raspberry Pi and HummingBoard being brought into the Internet of Things (IoT) are cause for concern. The question of how big a security threat such devices can pose, particularly when run with poorly secured RBOI while connected to the Internet, becomes an important one.

In order to address these concerns this paper proposes an analysis on a number of commonly used RBOI and their associated SBC to determine the potential security threat these devices could pose when turned into botnet slaves.

The rest of this paper is structured as follows: In Section 2 the goal and research questions are discussed, Section 3 further elaborates on the approach that was used in order to answer each question and Section 4 lists hypotheses and expected results. Section 5 details a literature study into more relevant combinations of SBCs and RBOIs followed by Section 6 where known vulnerabilities of the found combinations are discussed. Section 7 then serves to detail a case study into the scalability of SBC botnets. Section 8 and Section 9 contain the final conclusions and discussion of our findings.

2. GOAL AND RESEARCH QUESTIONS
The goal of the proposed research is to illustrate the potential power of an SBC botnet and the threat they might pose if their RBOI are not carefully configured.

In order to achieve this goal and to guide the research, the following main research question has been defined:

Q1) Is a Denial of Service attack executed by a Single Board Computer potent enough to consider building a botnet?

In order to better answer the main research question a number of subquestions have been formulated:
Q1.1) What are the most commonly used Readily Bootable OS Images, for what purpose are they used and what Single Board Computers do they run on?

Q1.2) What are commonly observed Readily Bootable OS Image vulnerabilities that can potentially pose a significant threat?

Q1.3) How strong are DDoS attacks executed by Single Board Computers running Readily Bootable OS Images?

3. APPROACH
In order to answer Q1.1 a literature study was conducted in order to research what SBC are currently available as well as the RBOI that are run on these devices. For this research we only considered devices with a significant market share, RBOI were selected based on whether or not the size of the user base was deemed large enough to be considered a potential threat. This helped us focus our attention on the more relevant combinations. SBC and RBOI that have an extremely small user base are less likely to become target of malware developers or other malicious parties and subsequently are less likely to be used in a botnet, making them less interesting for our research.

After we established a comprehensive list of the more commonly used RBOI and sorted it out by the number of users, it was important to research any known vulnerabilities that these images may have. For this purpose a background study and literature study were conducted. The background study focus lies on the communities behind the more popular images. Community forums and wikis were searched for vulnerabilities that the users were already aware of. The literature study then looked into common weaknesses found in community driven operating systems. If there were OS images in our established list with a large number of the vulnerabilities found in either of the studies it will be more likely that these can and will be exploited by attackers.

Thirdly in order to answer Q1.3 a case study was conducted using Raspberry Pi running the most relevant RBOI that was determined based on popularity and vendor recommendation. To research the potential of a DDoS attack that is carried out by a botnet consisting of these devices we measured generated traffic in a lab setup. In the experiment 5 devices attempted to take down a web server. The setup is shown in Figure 1. A single device attempted to perform normal operations with the server such as retrieving webpages or downloading files to see the effect of the attack for regular users. The traffic used to attack the server with for this case study was generated using several tools. A full listing of the tools used to generate benign and malicious traffic as well as the tools used to capture the traffic during the experiment can be found in section 7 which further details the case study.

4. HYPOTHESES AND EXPECTED RESULTS
Based on the information collected during the initial stages of this research’s proposal the following expectations have been formed:

1. it is expected that the Raspberry Pi will enjoy a proportionally high amount of popularity among users. It stands to reason that the popularity can be attributed to the strong effort the Raspberry Pi foundation has put into marketing it as an educational tool. Combined with user friendly tutorials and documentation the threshold for the novice user becomes extremely low making it the ideal all round SBC.

2. Since the Raspberry Pi foundation maintains its own Debian distribution, Raspbian, it is expected that many users of the Raspberry Pi will also use this officially supported RBOI.

3. The limiting factor for the attack power of a Raspberry Pi, when executing conventional DDoS attacks, is probably its network interface. As such the power of a single Raspberry Pi is limited but the strength of a DDoS attack will scale with the size of a botnet.

In the following section we will discuss the literature study that was conducted in order to answer questions Q1.1 and reflect on hypotheses 1 and 2.

5. LITERATURE STUDY
As a starting point for our literature study we looked at the SBC survey [6][7] held by LinuxGizmos and the Linux Foundation. The 2014 survey revealed to us a comprehensive list of SBC that showed signs of popularity among the 777 respondents.

As expected we found the Raspberry Pi (Model B specifically) at the top of the popularity list. The number two competitor BeagleBone (Black) is close behind but the other competitors lag far behind in these rankings.

Figure 2: Top 10 Open SBCs — May 2014.

After noting the results from the 2014 survey we compared them to the 2015 ones in order to determine if the Raspberry Pi popularity was short lived or further on the rise. The survey [16], featuring 1721 participants this time, showed a remarkable result beyond initial expectations. While the updated Raspberry Pi Model B+ now ranked lower than its competitor, the BeagleBone Black, the newer model Raspberry Pi 2 Model B immediately took first place further affirming the popularity of the Raspberry Pi family of SBCs.

Taking the results of both surveys [6][7] into account we continued our research by looking into the RBOI that could run
Figure 3: Top ten Hacker SBCs — May 2015.

Since it would prove to be hard to determine the exact popularity of specific OS images on each respective SBC we attempted to create an estimate by counting the number of threads and posts regarding a specific OS on the official community forums for each SBC. We then later looked up the official community forums for each OS and applied the same method for mentions of a specific SBC and aggregated the results together to make a final estimation. The results per SBC and RBOI can be found in Table 1.

Table 1. Post and topic count per SBC and OS.

<table>
<thead>
<tr>
<th>Device</th>
<th>OS</th>
<th>Posts</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raspberry Pi</td>
<td>Raspbian</td>
<td>3127</td>
<td>4390</td>
</tr>
<tr>
<td>Arch</td>
<td></td>
<td>929</td>
<td>702</td>
</tr>
<tr>
<td>Pidora/Fedora</td>
<td></td>
<td>1973</td>
<td>330</td>
</tr>
<tr>
<td>RISCOS</td>
<td></td>
<td>3567</td>
<td>294</td>
</tr>
<tr>
<td>Kodi(XBMC)</td>
<td></td>
<td>3154</td>
<td>1056</td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td>1021</td>
<td>1403</td>
</tr>
<tr>
<td>BeagleBone</td>
<td>Debian</td>
<td>3290</td>
<td>604</td>
</tr>
<tr>
<td></td>
<td>Android</td>
<td>708</td>
<td>147</td>
</tr>
<tr>
<td>Ubuntu</td>
<td></td>
<td>3726</td>
<td>647</td>
</tr>
</tbody>
</table>

Since both Raspbian and Pidora were both developed exclusively with the Raspberry Pi in mind we did not assume to find mention of these RBOI in community forums of other SBCs. In order to determine the popularity of certain SBCs from the OS perspective we grouped Raspbian with Debian and Pidora with Fedora respectively. In Table 2 we determined the number of posts and topics per SBC for each OS. The results also show a number of other SBC that were not found in the starting point surveys [6][7], since malware and exploits are usually made on a software level it is possible that certain RBOI that run on multiple SBCs enjoy a relatively large user base compared to Raspbian for example since Raspbian has been developed for the Raspberry Pi exclusively.

As originally expected, the data in Table 2 shows that the Raspberry Pi seems to be the most popular SBC for almost every single OS community. The conclusion of our literature study confirms hypothesis 1 by showing the prominence of the Raspberry Pi within the SBC community. Hypothesis 2 is also confirmed by showing Raspbian to be the most popular RBOI on the Pi subsequently answering Q1.1

6. KNOWN VULNERABILITIES

We try to answer Q1.2 in this next section by conducting further background research on the known vulnerabilities present within Raspbian and Debian. As a starting point for known vulnerabilities in the Debian OS we read the official hardening walkthrough [9]. A notable security vulnerability we found was the absence of support for position independent executables and Address space layout randomization (ASLR). Both features make certain types of attacks such as return-to-libc attacks [10] more difficult by having the system obscure related memory-addresses from attackers. If an attacker guesses the wrong memory-address to attack their attack will be hindered by having the attacking malware crash. The absence of these security measures make it much harder for a machine to defend against zero-day exploits.

Another possible Debian vulnerability we discovered was found while reading through the official Debian documentation [19]. According to the manual itself it is possible to include scripts as part of a package that the package management system will be run when a package is installed, updated or removed. Since the package management system will run with root privileges when the users decides to install a new package it essentially enables any package that comes bundled with a malicious script to take over an entire system through privilege escalation.

The most notable vulnerability that we encountered was found in the Raspberry OS, according to Raspbian documentation SSH has been enabled by default with a default username and password [16]. This makes Raspbian systems vulnerable from the moment they are connected to the Internet unless a user performs the installation offline and modifies the username and password combinations before connecting their Raspberry Pi to the Internet. As a matter of fact the default Raspbian password can already be found in password dictionaries online [8].

Another paper A Risk Assessment on Raspberry Pi using NIST Standards [21] ascertains through a thorough assessment of literature surrounding the Raspberry Pi and its documentation that the Pi is susceptible to at least 7 of the 10 worst vulnerabilities listed in the Open Web Application Security Project (OWASP) top-ten worst vulnerability list as seen below in Figure 4. Yet in spite of this the research as described in the
paper determined the overall security risk of the Raspberry PI in accordance with The National Institute of Standards and Technology (NIST) standards to be Moderate [21].

7. RASPBERRY PI CASE STUDY

In order to determine the potential danger of DDoS attacks executed by SBC running RBOI, we conducted a case study on a group of Raspberry Pi running Raspbian. As a setup for the case study we use a simple network topology as illustrated in Figure 1. where all but one Raspberry Pi is part of a botnet that is attempting to execute a DDoS attack on our server. During the early stages of our research we looked extensively at how to map the strength of an executed DDoS attack. Most of the works we looked at for reference often used single parameters to determine attack strength and while these parameters were useful in creating a general idea of the strength of the attack we believe that in order to fully understand the potential damage an attack can cause additional metrics are necessary. This theory is confirmed by J. Mirvovic in Measuring Denial-of-Service [14]. Mirvovic states that in order to capture the impact of a DoS attack more precisely, measuring DoS impact as a percentage of transactions that have not met their QoS requirements gives a better view compared to more traditional individual metrics such as duration, throughput or loss.

Another focal point in this case study is the scalability of the SBC botnets used in these attacks. In order to determine how well the strength of these botnets scale with the number of SBC we needed to ensure that all of the tests we run are reproducible. The percentage of failed transactions in the duration of our experiments are highly dependent on the traffic used, as such we needed a reliable way to generate the same benign traffic during each experiment to ensure we got similar results. Finally we can study the scalability of the SBC botnet by comparing the the results of the experiments when run with a various number of SBC's. Real-world scenarios often feature much larger scale botnets and infrastructure compared to the one used in this study but the setup used in these experiments should suffice to illustrate the points made in this paper and any metrics derived from the performed tests should prove scalable to larger setups.

In each experiment we measure the same metrics as Mirvovic [14]: (1) One-way delay, (2) request/response delay, (3) packet loss, (4) overall transaction duration and (5) delay variation(jitter). The mentioned DoS impact metrics are collected by using traces captured with tcpdump, (2) request/response delay and (4) overall transaction duration are collected using a trace on the sender side. (1) One-way delay, (3) packet loss and (5) delay variation(jitter) are measured by collecting traces from both the sender and receiver sides. The correlation used to obtain the metrics is made by matching the source IP, port numbers and packet identification from both traces. In order to ensure that correlation can be done effectively, the clocks of all computers are synchronised at the start of each experiment.

Once all of the impact metrics are collected we use them to objectively measure the Quality of Service (QoS) as perceived by the user. The perceived QoS varies for each individual and is largely based on the expectation of the user depending on the application they are using [5]. It stands to reason that because of this we measure QoS differently for each type of application. For interactive applications we use request/response delay to capture the degradation of QoS, media and video game applications use one-way delay, packet loss and jitter and finally transaction duration is used to measure the QoS for non-interactive tasks.

We derive the degrade in QoS for each failed transaction by finding a parameter that exceeded the QoS threshold of the corresponding application category and calculating the ratio of their difference and the threshold using the formula: QoS-degrade = \frac{d}{t} where d is the measured delay and t is the threshold value. If multiple parameters violate their threshold we choose the largest QoS-degrade. We take our threshold values directly from Mirkovic’s paper [14] but for the convenience of the reader we have copied the table with the corresponding values directly into Table 3. In experiments, we report the average of the QoS-degrade measures for transactions in the same application category. A value N of QoS-degrade means that the service of failed transactions was N times worse than the determined user tolerance level.

In order to establish a baseline we first generate legitimate traffic from our benign client to our server. The traffic will consist of: (1) HTTP and FTP traffic and (2) DNS and ICMP traffic. All real traffic is generated using real applications: scp for FTP, ping for ICMP, wget for HTTP and dig for DNS. We use Bonesi to simulate both UDP and TCP floods from large-scale bot networks. Our server will be running an Apache web server, bind DNS server and Linux Mint operating system. All legitimate traffic is generated throughout the duration of each experiment, after 60 seconds an attack is launched that will last another 60 seconds before ending.

### Table 3: Application categories and their QoS requirements [14].

<table>
<thead>
<tr>
<th>Category</th>
<th>One-way delay</th>
<th>Req/res delay</th>
<th>Loss</th>
<th>Dur.</th>
<th>Jitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>email</td>
<td>&lt; 30 s</td>
<td>whole, RTT &lt; 4 s</td>
<td>whole, RTT &lt; 4 s</td>
<td>&lt; 60 s</td>
<td>&lt; 300%</td>
</tr>
<tr>
<td>Web</td>
<td>&lt; 150 ms</td>
<td>any, RTT &lt; 4 s</td>
<td>any, RTT &lt; 10 s</td>
<td>&lt; 3 %</td>
<td>&lt; 300%</td>
</tr>
<tr>
<td>FTP</td>
<td>&lt; 500 ms</td>
<td>any, RTT &lt; 250 ms</td>
<td>any, RTT &lt; 4 s</td>
<td>&lt; 300%</td>
<td>&lt; 300%</td>
</tr>
<tr>
<td>Audio conv.</td>
<td>&lt; 150 ms</td>
<td>whole, RTT &lt; 4 s</td>
<td>whole, RTT &lt; 4 s</td>
<td>&lt; 3 %</td>
<td>&lt; 50 ms</td>
</tr>
<tr>
<td>Audio, stream</td>
<td>&lt; 10 s</td>
<td>whole, RTT &lt; 4 s</td>
<td>whole, RTT &lt; 4 s</td>
<td>&lt; 1 %</td>
<td>&lt; 50 ms</td>
</tr>
<tr>
<td>Video phone</td>
<td>&lt; 150 ms</td>
<td>whole, RTT &lt; 4 s</td>
<td>whole, RTT &lt; 4 s</td>
<td>&lt; 3 %</td>
<td>&lt; 50 ms</td>
</tr>
<tr>
<td>Video, stream</td>
<td>&lt; 10 s</td>
<td>whole, RTT &lt; 4 s</td>
<td>whole, RTT &lt; 4 s</td>
<td>&lt; 1 %</td>
<td>&lt; 50 ms</td>
</tr>
</tbody>
</table>

7.1 UDP Flood

In the first experiment we generate an UDP flood that eventually overwhelms the server. The initial test runs of the experiment revealed that the maximum output a single Raspberry Pi 2 can produce in our setup over UDP is roughly equal to 16Mbps. This maximum output was determined by evaluating the CPU consumption using a resource monitor, top, and at the same time evaluating the produced output volume with the bandwidth monitor bmon. Since both the switch and the server can handle 100Mbps worth of traffic we limited the server’s link capacity to 20Mbps to create a bottleneck. Figure 5 shows the percentage of failed transactions for each executed attack. The x-axis shows the number of Raspberry Pi’s that
were used to execute the DDoS attack, the column height denotes the average of three test runs. Because of complications with the setup we had to drop the FTP traffic and were only able to include the pft measurements for the HTTP, ICMP and DNS application categories. We do however believe that this subset of Mirkovic’s measurements should be sufficient to provide an indication to the DDoS capabilities of SBC botnets. Both DNS and ICMP transmit small single package requests. While DNS still retransmits 3 times, ICMP never retransmits lost packets. In contrast HTTP is far more resilient to smaller attacks since it aggressively retransmits lost packets and has a large delay threshold of 10s.

8. CONCLUSIONS
The experiment shows that potential for a botnet based on SBCs exists. While our setup is simplified compared to a real-world scenario the increase in computing power gained from having a significant higher number of SBCs should more than make up for the limitations imposed by our setup. While even a simple botnet would already be sufficient to deny the HTTP service of a simple web server, a more sizeable botnet would be able to fully deny all three of the application categories that we tested for. In the end DDoS attacks remain an incredibly complex phenomenon that poses a significant threat to the current network infrastructure. With the number of devices that are continuously being added to an increasingly larger Internet of Things it becomes a serious question as to how the society at large should deal with security of their devices.

9. DISCUSSION
Earlier in this paper we mentioned the rise of the IoT and ever growing number of devices becoming connected to it. The number of critical infrastructure controlled and managed by SCADA systems is also being increased by governments and corporations that wish to cut budget spending on personnel or otherwise increase their control of said systems. Consolidation and shared processes or systems between the public and SCADA environments may increase security and availability risks to the SCADA environment. The increased attack surface creates a number of potential dangers for critical infrastructure controls that may not pose a significant threat in regular corporate networks but could prove to be catastrophic in SCADA environments. While examples such as Stuxnet have shown just how vulnerable SCADA systems are when they are targeted. Other forms of malware could potentially wreak havoc when they manage to enter SCADA environments through their connected corporate networks without ever having been made with the intention of doing so. In fact a number of examples already show that un-targeted attacks can cause visible damage to the infrastructure managed by these SCADA systems. In 2003 the third largest railroad company in The United States, CSX, suffered a 2-hour signal outage causing delays for both transport and commuter trains [12]. Because of the sensitivity of the systems controlled by SCADA systems even the slightest delay in communication can cause serious damage as is evidenced in an internal video of the US Department of Homeland Security where they demonstrate how they overload a reactor turbine to the point of self-destruction by use of a network based attack [13]. The concerns raised in the introduction still stand and are further reinforced by the findings in this paper. The vulnerabilities of SCADA systems and the potential danger coming from a growing Internet of Things urge me to stress the point that governments should focus their efforts on improving the security of these systems before bringing more critical infrastructure under the government of inter connected SCADA systems.

10. FUTURE WORK
Due to time and resource constraints this research mainly focused on executing DDoS attack using a relatively small number of Raspberry Pi’s. To further investigate how the strength of a botnet consisting of SBCs scales with the number of devices, a larger number of Raspberry Pi’s in the
experiments could provide more interesting results. Especially when it comes to the TCP flood a larger number of devices would be required to further investigate the scaling of TCP flood attacks on SBCs.

Complications with the setup have lead to the FTP protocol being excluded from experiments. An additional complication with Bonesi running a TCP flood based on HTTP-GET request caused the use of a different attack tool for the TCP flood experiment creating a potential drift in sample data. In future work it would benefit the experiment to have a better configured setup to provide a more standardised output. Additionally it would be interesting to see how well the SBCs perform when executing different types of attacks.

Future research should also take care to implement to full range of Mirkovic’s suggested metrics to create an accurate view on the DDoS potential of botnets.

Finally the small amount of runs for each of the experiments also create an unstable average for the number of failed transactions metric we measured. Future research would be able to provide for a more realistic view on the failure rate by using an increased number of runs.

11. REFERENCES