Teaching Security of Internet of Things in Using RaspberryPi

Oliver Nichols
University of Tennessee at Chattanooga, mpl934@mocs.utc.edu

Li Yang
University of Tennessee at Chattanooga, lyang03@gmail.com

Xiaohong Yuan
North Carolina A& T State University, xhyuan@ncat.edu

Abstract
The Internet of Things (IoTs) is becoming a reality in today's society. The IoTs can find its application in multiple domains including healthcare, critical infrastructure, transportation, and home and personal use. It is important to teach students importance and techniques that are essential in protecting IoTs. We design a series of hands-on labs in a smart home setting, which can exercise attack and protection of IoTs. Our hands-on labs use a Raspberry Pi and several diverse smart things that communicate through Z-Wave technology. Using this environment, students can operate a home automation system and learn security concepts by performing these labs. These labs demonstrate several fundamental security concepts and techniques that can be adopted in security curricula. Students are expected to understand and master how to implement various attacks, design and implement defenses to these attacks, and explore security solutions of Internet of Things in a Smart Home application.

Disciplines
Curriculum and Instruction | Information Security | Management Information Systems | Technology and Innovation

INTRODUCTION

The "Internet of Things" (IoT) is a concept aimed at making our devices, gadget, appliance and transports all interconnected in the name of convenience and Big Data – whether businesses want additional data on their customers or consumers want to control their appliances through their smartphones. The architecture of the Internet of Things actually comprises of users who can perform device management or business processes. The IoT has several definitions. The IoT Special Interest Group defines the Internet of Things as a "revolution already under way that is seeing a growing number of internet enabled devices that can network and communicate with each other and with other web-enabled gadgets. IoT refers to a state where Things (e.g. objects, environments, vehicles, and clothing) will have more and more information associated with them and may have the ability to sense, communicate, network, and produce new information, becoming an integral part of the internet." (ITSIG, 2013) Things, as physical devices that touch the real world in multiple domains, can be sensors – optical, thermal, mechanical, and so on – that measure the physical states of houses, machines, or people.

The Internet of Things are pervasive around us today. Examples include smart health application where wearable devices are used to monitor user health and behavior information so that they can learn more about their health and improve self-fitness and wellness. Smart home is another emerging application where users can remotely control their home environment and use sensing devices for various tasks, such as detecting a break-in, changing the thermostat, and adjusting the lights. A variety of things are available as off shelf products available at local stores, such as Wal-Mart and BestBuy.

The Internet of Things has been gaining momentum over the past few years. According to Postscapes, the funding for the Internet of Things has been millions of dollars since the year 2010 (Postscapes, 2014). What could companies hope to achieve with millions of dollars invested into the Internet of Things? Two useful applications for the Internet of Things may include improving efficiency for consumers and businesses.

For businesses, the Internet of Things has the potential to reduce costs and promote higher sales. One of the goals for the Internet of Things is to make deployment of devices cheap. Making the deployment of devices inexpensive could promote businesses and consumers to adopt the Internet of Things. For businesses, this could mean low expenses and high returns of useful information.

Smart home applications improve the efficiency and reduce costs for users. For example, a user Bob is at work, and he forgot to lock the house door. With a Smart Home door lock, he can lock his doors from work. He could also save heating and cooling costs through using a smart thermostat remotely. As the Internet of Things grows and broadens, increased number of applications will rise to provide energy, time, and cost efficiency for consumers.

The emerging popularity of IoT also draws the attention of cyber-attackers. If a cyber-attacker is able to break into one of such system, they potentially can harm thousands of people with little effort. In a recent interview with the Harvard Business Review, cryptographer and security expert Bruce Schneier said that while website security is a major issue, the Internet of Things will prove much more difficult to manage. "These are devices that are made cheaply with very low margins, and the companies that make them don't have the expertise to secure them. Heartbleed (Heartbleed, 2016) would have been much worse in a world of Internet enabled thermostats, refrigerators, cars, and everything else, and that's the world we're headed toward" he commented. A hacker exploiting the IoT to maliciously turn off home thermostat would be an annoyance. But if that hacker could manipulate furnace to trigger an explosion, or intercept sensor data to identify when children are home alone, security would take on an entirely different imperative. Therefore, resource and information present in the distributed systems in Internet of Things (IoT) must be protected.

We use smart home as a study case in this paper. Ray (Ray, 2013) wrote a brief article online and summarized a Black Hat 2013 presentation where researchers were able to gain access to a Z-Wave door lock, part of things in a smart home. This type of access would show no signs of forced entry into a home. These types of attacks demonstrate the importance and significance of security in the Internet of Things. This type of vulnerability comes from poor design and lack of security expertise in the manufacturer of the door lock. Other applications, such as SmartThings (SmartThings, 2016), use the cloud to control and synchronize data. Connecting the Internet of Things into the cloud could create additional new attack vectors and affect other cloud consumers.

This paper presents a series of experiential hands-on labs designed to illustrate how to secure IoT in a real-world deployed smart home setting. The goal is to provide a real-world learning and exploration environment for students and prepare them with security skills and knowledge in the field of IoT A compromised Smart Home can result in serious consequences. The attacker could gain access to sensors, door locks, cameras, and other similar items. Access to these items could threaten the privacy and security of the home. These ideas can be expanded to the broader sense of the Internet of Things. As the Internet of Things finds new applications, it is crucial that security is implemented in the design of the devices and software. The hands-on labs demonstrate five possible intrusions targeting at communication, protocol design, and software levels.

SETTING UP EXPERIMENT ENVIRONMENT

We set up a Smart Home environment using a Raspberry Pi, Razberry Daughter Board, and Z-Wave sensors. With these elements, students can learn about security and how it is significant in the application of Internet of Things.

Choosing Devices and Protocols

The Raspberry Pi was chosen because it is a small device that can run a full Linux operating system on ARM architecture; its power consumption is similar to charging a phone; and the Raspberry Pi has gained much popularity and support by the community.

The Razberry Daughter Board was chosen so that Z-Wave functionality could be integrated into the Raspberry Pi. The Razberry Daughter Board comes with software to provide home automation. The board is specifically designed to be used on a Raspberry Pi.

Z-Wave is one of the newer protocols for low powered wireless devices. Z-Wave is closed source, but it is also interoperable across brands. In the end, interoperability made Z-Wave the protocol of choice for this project.

The Environment

The underlying operating system (OS) used on the Raspberry Pi is Raspbian. It is a version of Debian designed for the Raspberry Pi. It is the recommended OS by the new out-of-the-box software (NOOBS) [RaspberryPi, 2016].

The environment uses Z-Wave devices including Aeotec multisensors, Remotec Plug-in Dimmer, and a GE outlet control module.

EXPERIENTIAL HANDS-ON LABS

The software that implements the automation and talks to devices is the Razberry software. It can connect to the IP camera and wireless Z-Wave devices using the Razberry Daughter Board. More information about Razberry can be found at (Z-Wave, 2016).

We designed experiential hands-on labs that are used to demonstrate relevant security concepts in the context of Smart Homes, which is an application of the Internet of Things. These labs show the vulnerabilities and protection of a smart home application. Raspberry Pi was used in these labs to connect with diverse smart things because Raspberry Pi is inexpensive which suits education environment, and it has large user group. The operating system Kali, the newer version of Backtrack, was also used because it has penetration tools needed for the labs.

Students will be challenged with threats and intrusions to the smart home setting, and then they will find or speculate on ways to defend against these attacks. These tasks will help students to understand hacking and defense in smart home applications.

Sniffing of IoT and Its Countermeasures

Sniffing is a form of passive attack to learn about the communication between two computers. In the wrong hands, it can be used to discover passwords, analyze traffic, or learn more about a target system. Sniffing is a fundamental attack students need to learn about. This lab allows students to learn how to apply it, understand how to defend against it, and see the resulting consequences.

Experiment Requirements

The software Wireshark will be used to sniff packets. Wireshark requires a network interface to capture packets. The attacking machine using Wireshark should be on the same local network as the Raspberry Pi. Make sure that the devices are not connected to a switch, otherwise a Man-in-the-Middle attack would be required.

Remote Access needs to be enabled in Razberry for this exercise. This can be done in the Z-Wave Configuration by using the browser to access the IP address of the Raspberry Pi at port 8083 (e.g. 192.168.0.106:8083). The address find.zwave.me can be used in the browser to discover the address of the Raspberry Pi. In the configuration page, remote access can be enabled. This page also displays the ID for the Raspberry Pi and allows the user to change the password. Figure 1 shows an example of lab environment with Raspberry Pi and sensor nodes (Z-wave nodes).

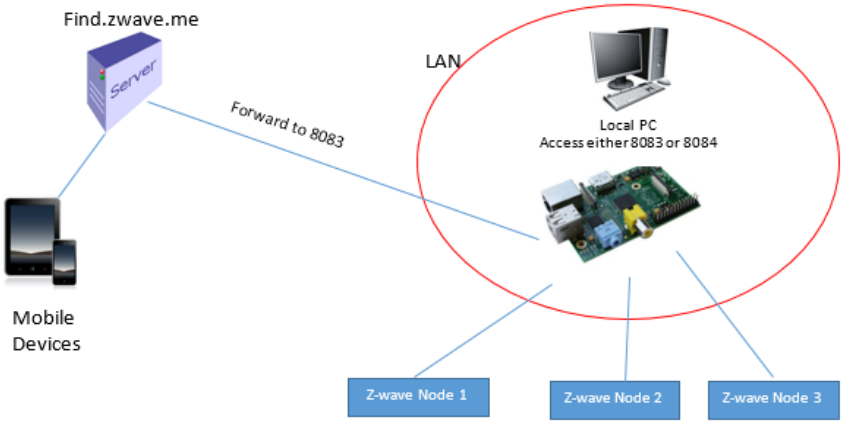

Figure 1: The Architecture of IoT in a Smart Home Setting

Lab Description

To begin, the legitimate user will use find.zwave.me to log into his Raspberry Pi remotely. The user was not very careful and did not notice that find.zwave.me was currently set to "Not a secure connection." Using Wireshark, a student will sniff the packets on the network that the Raspberry Pi is receiving. The student should be able to discover the ID and password because the communication is unencrypted.

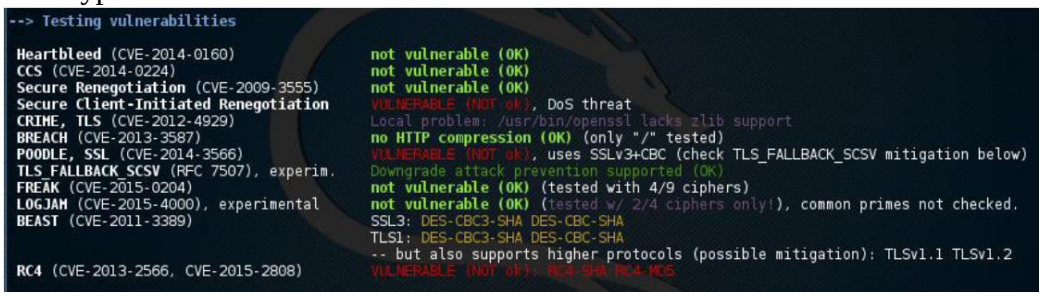

Figure 2: Security Analysis of Remote Access

As can be seen above, the test results show a possible DoS threat (CVE-2009-3555) and the SSL implementation allows for use of RC4; a potential threat, but there are no easy-to-find "proof of concepts" in regards to CVE-2011-3389.

Students are then instructed to analyze security given access to local area network (LAN) where Raspberry Pi is connected with. Students are allowed to use sniffing tools such as Wireshark to sniff LAN traffic. Logging in locally shows no form of encryption is used. This leaves the system vulnerable to malicious users who gain access to the LAN. Accessing port 8084 (i.e. the "settings" application) requires no authentication. Furthermore, accessing the web application itself leaves anybody on the LAN to observe plaintext credentials, and the ability to perform a cookie hijacking attack. Observing the traffic capture, we can see the credentials are passed in JSON format and in plaintext as shown in Figure 3.

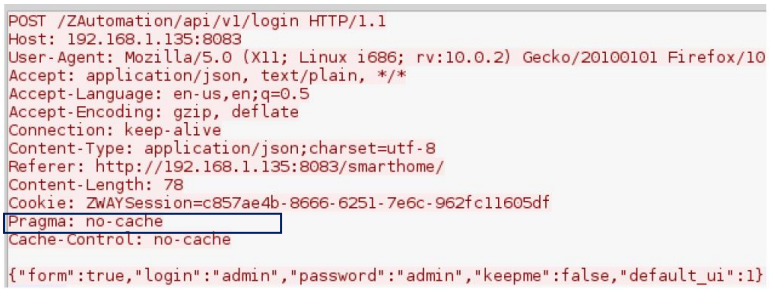

Figure 3: Security Analysis of Local Access

The user this time is more careful on the find.zwave.me and toggles the option "Not a secure connection" to a secure connection. The user should take note of the changes that occurred. Specifically, in the URL, the address should change from a HTTP connection to a HTTPS connection. Using Wireshark again, a student should observe the packets and be unable to discover the ID or password due to encryption of TLS.

Lab Activities

Students are first instructed to analyze the security in remote access which includes communication between the mobile device and find.zwave.me, and communication between find.zwave.me and Raspberry Pi. Students analyzed the web application hosted on the Raspberry Pi. The connection from the Mobile device to the server used TLS 1.2. The certificate appears to be valid, and no issues were reported using Firefox, Internet Explorer, or Chrome. Traffic between the server and the Raspberry Pi also used TLS. Vulnerabilities for TLS were tested on the Raspberry Pi's implementation using a test script which can be found at https://testssl.sh/testssl.sh. The script did not discover serious vulnerabilities as login username and password of Raspberry Pi was not discovered. Figure 2 shows the results of the "SSL" test.

Learning Outcomes

Students are able to use sniffing tools to analyze communication from both remote and local access. They can understand and analyze various communication protocols and discover potential vulnerabilities. Students can discover how easily a username and password can be apprehended on an insecure connection. Students should note that prevention for passive attacks, such as sniffing are much easier to prevent than detect. Sniffing can be effectively prevented through the use of encryption.

Session Hijacking and Replay Attacks

Lab Description

Session ID or cookie ID is a unique identifier used to determine communications between two parties. Session hijacking is to first acquire the session ID and then combine with replay attacks to impersonate one party of the communication.

Lab Activities

Students are first asked to discover the vulnerabilities in the communication between the Raspberry Pi and the sensors, for example, how to acquire the session ID and the lifetime of the session ID. Figure 4 shows that the session ID between Raspberry Pi and the smart switch will be alive even after the Raspberry Pi's administrator logs off.

```
GET /ZAutomation/api/v1/devices?since=1448825740 HTTP/1.1
```

Figure 4: Session ID between Rasberry Pi and Smart Switch

Students are then asked to use the same session ID to replay the control message from the Raspberry Pi to the smart switches. Students are able to change "on" and "off" easily by replaying the control message with its session ID.

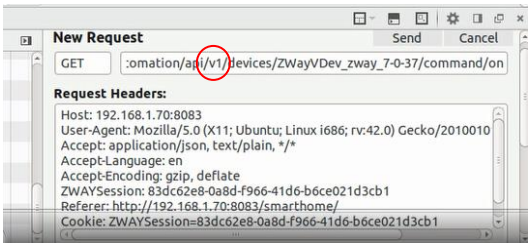

Figure 5: Replay Attack to Respberry Pi and its Linked Smart Switch

Learning Outcomes

Students are able to understand session hijacking and replay attack through working with a system formed by a Rasberry Pi, a mobile device, and multiple sensors and actuators. Students are able to intercept message, capture session ID, and replay control message.

ARP Cache Poisoning Attack

Man-in-the-Middle (MitM) attacks can be achieved by using ARP cache poisoning on a local network. ARP cache poisoning is an achievable task due to the fact that no authentication mechanism exists for ARP.

Experiment Requirements

The software ettercap can be used to implement an ARP cache poisoning attack between the Raspberry Pi and another machine. If the user is accessing the Raspberry Pi remotely, the attack will be conducted against the router and the Raspberry Pi. If the user is accessing the Raspberry Pi locally, the attack can be conducted directly on both the Raspberry Pi and the user's machine.

The software nmap can be useful for discovering the Raspberry Pi on a local area network (LAN). In a real world application, the attacker will not know the Raspberry Pi's IP address and will need to find it.

Lab Description

To begin the attack, the students will need to know the address of the Raspberry Pi and the machine attempting communication with the Raspberry Pi. This information can be given to the students or discovered using Wireshark, nmap, or another tool.

Table 1. Mapping between IP and MAC addresses

Host Name	IP	MAC Address
Host-A PC	192.168.1.70	b8:27:eb:40:39:77
Attacking PC	192.168.1.108	00:14:22:30:a3:6d
Victim Raspberry Pi	192.168.1.212	00:12:3f:54:73:f4

Using both addresses, the student will use ettercap to perform an ARP cache poisoning attack. The attack can be designed to watch all traffic that passes or talk directly to the user. The traffic between Host-A and Raspberry Pi will all be directed to the attacker's machine.

2.3.3 Lab Activities

Students are first instructed to modify ARP table so that Host-A's IP address will be mapped to the MAC address of the attacking PC as seen in Figure 6.

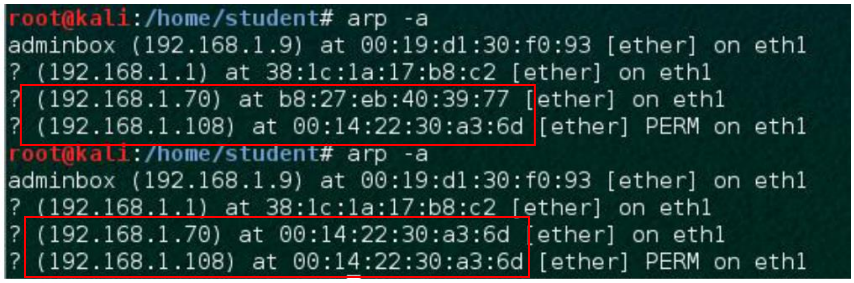

Figure 6: Before and After Cache Poisoing

Students will then confirm packet vulnerability with the assistance of Wireshark or another equivalent packet-sniffing software. Students logged into the Raspberry Pi (Host-B) from the Host-A computer. As the password is sent in plaintext, sniffing shows Raspberry Pi's login password again as shown in Figure 7.

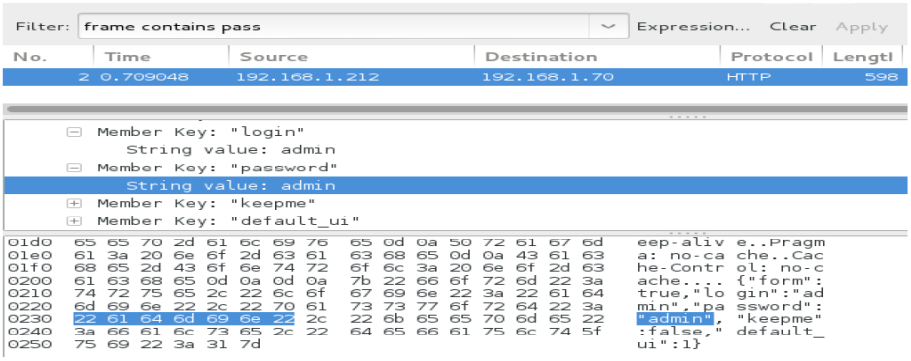

Figure 7: Password Revealed by an ARP Cache Poisoning Attack

Learning Outcomes

In this exercise, students will learn how to execute an ARP poisoning attack. Students will see how easily this can be done using the ettercap tool. Students should take note that ARP poisoning can only be accomplished on the local network. This increases the need for only allowed trusted devices on a local network and dividing a network to isolate resources.

Several methods exist for ARP poisoning detection, but no perfect system has been developed. The MAC addresses and corresponding IP addresses can be manually configured on each machine, but ARP updates must be disabled. The important aspect of this lab exercise is for the student to understand the attack and use critical thinking to understand several solutions and why they are not perfect.

ARM Stack Exploits and Security Measures

The Raspberry Pi B+ uses an ARM11 processor which is based on the ARMv6 architecture. Most mobile devices use ARM because it dissipates less heat and uses less energy than its major competitors. Given the processor is at the very center of all operations of the system, it is important to understand the potential security vulnerabilities at this level of operation.

Lab Description

A Stack is a set of registers or memory reserved for programs. Usually, a program consists of many routines to run in a sequential order. One routine "calls" another routine, and we refer to this relationship as a "caller-callee" relationship. Prior to the callee beginning its execution, the caller saves the address of the next caller instruction; and, after the callee completes, the address is used to reference the caller and execution continues in the caller routine. A buffer overflow (BoF) attack is to supply enough data to overfill the buffer, and write malicious data into an area that can be directly to the Link Register.

Lab Activities

Students are first asked to write a program called "overflow_exploit" with a typical Buffer Overflow (BoF) program called "vulnerable_callee" that copies the user-supplied parameter into a buffer with fixed size as shown in Figure 8. Students then are instructed to disassemble the code in RasberryPi and discover starting memory address of the "*vulnerable_callee*" as "0x00008498" in Figure 9. After determining the distance between the buffer and the stored return address, students will supply the beginning address of the overflow_exploit function to the return address of the vulnerable_callee function. Figure 10 shows the success of the attack where "*I should not be called*" was printed.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  void overflow_exploit() {
5      puts("I should not be called");
6      exit(0);
7  }
8
9  void vulnerable_callee(char *arg) {
10     char buff[10];
11     strcpy(buff, arg);
12 }
13
14 int main(int argc, char *argv[]) {
15     vulnerable_callee(argv[1]);
16     return 0;
17 }
```

Figure 8: A Buffer-over-flow Program

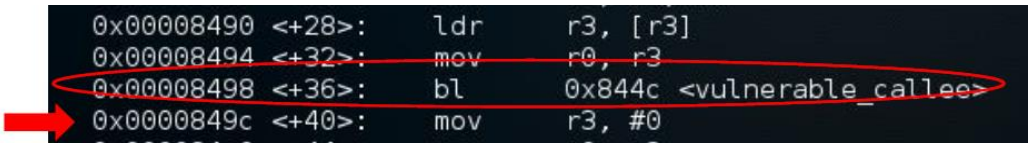

Figure 9: Starting Memory Address of vulnerable_callee

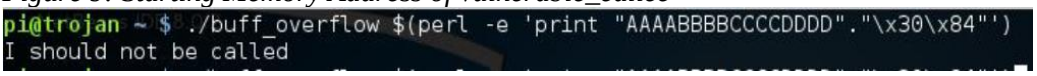

Figure 10: Success BoF attack

Students are also challenged to discover and exercise ways to defend again BoF attacks. Solutions include GCC's Stack Smashing Protection (SSP), Memory Protection using No Execute which is also refered as XN (eXecute Never since ARMv6).

Learning Outcomes

Students are able to understand the benefits and vulnerabilities of ARM architecture, a popular one in mobile devices. Students are able to write both vulnerable and secure programs, and also know various techniques to defend against BoF and related stack overflow attacks.

EVALUATION AND DISCUSSION

These labs are designed to demonstrate attacks and security practices; and show the significance of security in the Internet of Things. The attacks are designed to work against TCP/IP and the Razberry software.

These labs do not directly address Z-Wave because it is closed source and resources on Z-Wave are limited. Accessing Z-Wave packets for sniffing and injection requires specific hardware and knowledge. Some research on Z-Wave has been done (Fouladi, B. & and Ghanoun, 2013).

CONCLUSION AND FUTURE WORK

This paper discusses several labs based on Raspberry Pi, mobile devices, and smart sensors in a smart home setting. Students learn how to implement several fundamental attacks, and discover defense solutions. Many of these attacks are relevant and useful for a security skillset.

Future work may include adding additional attacks for students to study, such as attacks against the IP camera. Possible routes to broaden this lab are to include other protocols on the Raspberry Pi or acquire devices that allow Z-Wave analysis. Other protocols to include might be ZigBee or Bluetooth Low Energy. Both of these protocols are popular and would be beneficial for students to interact with. Adding additional protocols might require the use of different software. Razberry is specifically designed to work with Z-Wave products. These methods involve the use of a Raspberry Pi. Working with a Raspberry Pi can be a fruitful learning experience. Alternatives to the Raspberry Pi could include acquiring a hub for a Smart Home, such as the SmartThings hub. Depending on the type of hub that is acquired, a security analysis can be conducted on various elements: protocols, smartphone app, and hub software. It is likely that only attacks can be demonstrated on hub products. A Raspberry Pi might show an advantage over these devices because students can implement defenses for various attacks.

ACKNOWLEDGEMENT

Chris Bonham, Paul Bond, Wayne Simpson, Michael Crow have contributed greatly in developing and implementing hands-on labs discussed in this paper. This work is based in part upon work supported by the National Science Foundation under Grant Numbers DGE-1241651 and DGE-0551296. Any opinions, findings, and conclusions or recommendations expressed in this work are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

1. Postscapes (2014). Internet of Things Investments. DOI=http://postscapes.com/internet-of-things-investment.
2. SmartThings (2016). DOI=http://www.smartthings.com.
3. Fouladi, B. & and Ghanoun (2013), S., Security Evaluation of the Z-Wave Wireless Protocol, Black Hat USA.
4. Internet of Things Special Interest Group [ITSIG] (2013). Internet of Things (IoT) and Machine to Machine Communications (M2M) Challenges and Opportunities: Final paper May 2013. DOI=https://connect.innovateuk.org/documents/3077922/3726367/IoT+Challenges,%20final+paper,%20April+2013.pdf/38cc8448-6f8f-4f54-b8fd-3babed877d1a
5. Ray, B. (2013). REVEALED: Simple ‘open sesame’ to unlock your HOME by radiowave. DOI=http://www.theregister.co.uk/2013/08/13/wave_goodbye_to_security_with_zwave/

6. The Heartbleed Bug (2016), URL: http://heartbleed.com/.
7. RaspberryPi (2016). URL: www.raspberrypi.org, retrieved August, 2016.
8. Z-wave (2016). URL: razberry.z-wave.me, retrieved August 2016.