Network Security Practical Concepts, Importance, and Potential Implications

Kuo-pao Yang, Ray Dejean, Cory Clapp, Robert Banks, Daniela Raygadas, Isabella Bendanas
Computer Science and Industrial Technology Department
Southeastern Louisiana University
Hammond, LA 70402 USA

***Abstract*—This paper is to increase awareness of network security. Most programmers have a very limited amount of knowledge of the basics of network security. This paper illustrates a brief look at security at different levels of the Open Systems Interconnection (OSI) model, provides a practical application of common security related issues, demonstrations the implications regarding to hacking, and discusses a few methods of protecting sensitive data. In the hands-on application, programmers can gain knowledge about the fundamentals of network security and protect themselves from potential hacker vulnerabilities.**

Keywords—network security; cryptography; capture the flag; SQL injection; MD5 hash

I. Introduction

Network security is one of the fastest growing occupations in computer and information technology. It is important to understand the threats associated with accessing the internet for use in business, personal, or commerce settings. Hacking has become a very lucrative model for cybercriminals as more sensitive data is being digitized. The business of hacking has had rapid growth within recent years affecting over 1.5 million victims per day [1]. But, most programmers responsible for developing software are unaware of the concerns and implications surrounding basic security principles and ideas. The focus of software development is ease of use which is an inverse function to security. Therefore, the more secure an application becomes the harder the application becomes to use. The user education is not happening rapidly enough to combat the evolving nature of security [2].

The initial design of the Open Systems Interconnection (OSI) model that represents communication functions between systems does not incorporate security. The core principles of how computers communicate with each other over a network have not changed since the early 1980s [3]. It becomes a complex factor to secure data as more businesses, governments, and individuals store data through applications that connect with the internet. This lack of complexity in the model made it an easier integration into computer network communications. The OSI model was not required to secure data since it was not used to transport sensitive information at that time. The common protocols at each layer have no security as they were intended as a mechanism of transportation for encapsulating and decapsulating of data.

Security has two models to implement most the time. The first model is to make security programmers invisible to users. The problem of this model is that users will not be notified even if security is not there. Also, it does not inform users that security is not working properly. The second model is to have security warnings associated with everything users do. The problem of this model is that users are trained to ignore these warnings in local intranet sites and these habits transfer to outside local sites on the internet. The users disregard blatant warnings and proceed to conduct business since they are used to ignore them.

This project provides a solution to expose programmers in controlled settings on how to gain information using certain built-in features of network communications. This idea will help programmers understand the importance of security. Once programmers understand, they can apply precautions in dealing with data that connects with the internet. This model of applied knowledge must coexist with the efforts of securing unsecured protocols and proper security training on a more consistent basis [4].

This paper presents a series of controlled events and how they interrelate with security. It also introduces the problem and discusses hardware and software approaches. These tools are used to further develop the relation between user education and security measures. This paper also includes methods in which test scenarios are set up. Finally, conclusions are presented and issues for future research are discussed as well as the implications associated with malicious use of this work.

II. Related Work

Several projects have pioneered the use of offensive techniques as teaching concepts called Wargames, which have a long tradition among security enthusiasts [5]. In Wargames, the organizer creates a set of challenging scenarios of increasing difficulty which must be solved by the participants. Challenges usually are modeled somewhat after the problems an attacker faces when attempting a system penetration. Slightly more competitive than Wargames are Capture-the-Flag (CTF) or Deathmatch contests where teams battle against each other over the control of a network. The department of Computer Science at Texas A & M University offers an advanced networks and security research in which participants can get practical experience with defense and attack strategies in an isolated environment [6]. They use their own separate topology from an actual network in which it is interconnected by means of a router and hosts attached to two switches.

Currently, no similar projects locally exist at the level of competition that keeps up with the rapid changes in the security field. This project brings up the current network security standards and hopes to keep building on these efforts to create a lasting event that coexists with the academic side of computer science. This project brings a basic understanding of the network layout in which competitors will learn as they go.

By building the foundation of the effort to integrate security in the field of computer science, this project expects to accomplish a lasting event that can be built up into a more sophisticated activity. The entire controlled source code is open to the current academic base to add the general level of knowledge crucial for future generations of computer scientists. The integrated network allows users to accomplish tasks in an environment that can be easily rebuilt without fear of repercussions.

Formal ethical standards can be established for the computer science majors interested in network and security as this branch has a lot of implied hesitancy and a bad stigma [7]. By showing these techniques in which anonymous users can gain information about the systems, the importance of maintaining ethics can be emphasized. The implications should assist the process of designing applications to enforce proper security.

III. IMPLEMENTATION

This project sets up a small scale local area network between host servers in the Raspberry Pi's [8]. This project implements a switch that grabs information from network addresses and routing local information between two Raspberry Pi's. In Fig. 1, the Raspberry Pi comes with one 1.2GHz 64-bit quad-core ARM8 CPU, 1GB of RAM, 802.11n Wireless LAN, Ethernet Port, Micro SD card slot, HDMI port, and VideoCore IV 3D graphics core. It also comes with a NOOBS operating system installer that allows to select either from the current list or to get an operating system from the internet. This project uses the Raspbian operating system which is very similar to Ubuntu.

In software consideration, computer scientists are always adept at some programming languages. The biggest learning curve is how the security relates to the software. This project sets up four jeopardy-style security challenge events:

1. Find the public / private key
2. Exploit the weakness in a program
3. Gain permanent access to the server
4. Perform a rainbow table attack against an MD5 hash

In the first task, a public/private key pair, which uses asymmetric cryptography, is created. Asymmetric cryptography utilizes mathematical problems that currently admit no efficient solution such as elliptic curve relationships. A public key encryption attempts to ensure confidentiality that the person decides to encrypt the messages. A particular public key is authentic which is addressed by a public key infrastructure or a web of trust. The public key infrastructure sets roles, policies, and procedures needed to create, manage, distribute, use, store, and revoke digital signatures and manage public key encryption. A public key certificate includes information about the key, the owner's identity, and a digital signature of an entity that has verified the certificates contents are correct. These are signed by a third-party authority that validates a trust between the parties.

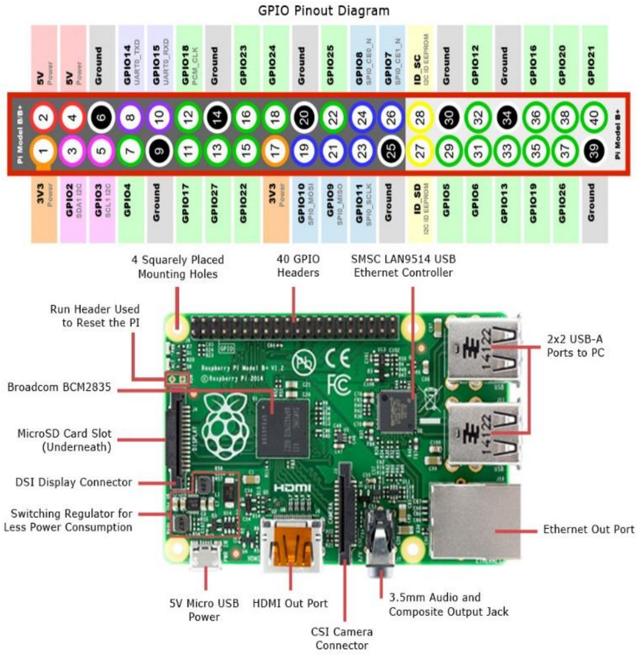
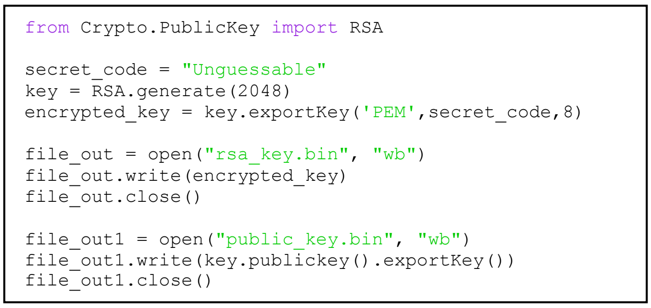


Fig. 1. *Raspberry Pi*

A created private key should be stored somewhere safe and can be used for decryption. By default, a public/private key pair is created in Linux and saved its location to a folder called .ssh. The contestants can search the key pair from root level to know the default location of where these files are saved and the naming convention used. To thwart the default knowledge of key pairs, the key is generated by using Python program shown in Fig. 2. The file name is related but not identified on the first glance to its contents.



```
from Crypto.PublicKey import RSA

secret_code = "Unguessable"
key = RSA.generate(2048)
encrypted_key = key.exportKey('PEM',secret_code,8)

file_out = open("rsa_key.bin", "wb")
file_out.write(encrypted_key)
file_out.close()

file_out1 = open("public_key.bin", "wb")
file_out1.write(key.publickey().exportKey())
file_out1.close()
```

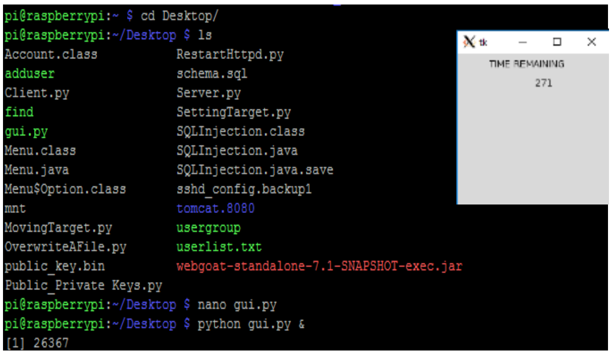
Fig. 2. *Python Code for Public and Private Key*

The contestants are required to find the public/private key. The file for the public/private key pair is moved to a random location under the root directory (‘/’) of Linux. The contestants need to be familiar with Linux commands and a small knowledge of what to look for. The contestants are not able to view or modify any code, but they are given hints about its content and default naming scheme. In Fig. 3, this Python program accomplishes the task by using shutil to access the operating system. The shutil module offers a number of high level operations on files. It invokes commands on the shell, and a random choice of folders under the root directory.

```
import random
import shutil
import os
source = '/home/pi/Desktop/rsa_key.bin'
directory = os.listdir('/')
if(os.path.isfile(source)):
   shutil.move(source, '/' + random.choice(directory)
+ '/')
```

Fig. 3. *Moving to a Random Directory*

During this task, the contestant will be given a time limit in the form of a Python Graphical User Interface (GUI) using the programing utility Tkinter. The Tkinter module is the standard Python interface to the default GUI toolkit. Fig. 4 shows a demonstration of a running program using the Ampersand Operator (&), which runs the commands in the background allowing contestants to process multiple commands while the task executes.



```
pi@raspberrypi:~ $ cd Desktop/
pi@raspberrypi:~/Desktop $ ls
Account.class           RestartHttpd.py
adduser                 schema.sql
Client.py               Server.py
find                    SettingTarget.py
gui.py                  SQLInjection.class
Menu.class              SQLInjection.java
Menu.java               SQLInjection.java.save
Menu$Option.class       sshd_config.backup1
mnt                     tomcat.8080
MovingTarget.py         usergroup
OverwriteAFile.py       userlist.txt
public_key.bin          webgoat-standalone-7.1-SNAPSHOT-exec.jar
Public_Private Keys.py
pi@raspberrypi:~/Desktop $ nano gui.py
pi@raspberrypi:~/Desktop $ python gui.py &
[1] 26367
```

Fig. 4. *Time Limit Graphical User Interface*

In Fig. 5, the Python program creates a popup box with the time remaining and then exits the GUI program after the countdown finishes.

```
import tkinter as tk
import time
def destroy():
   root.destroy()
def countdown(count):
   Title['text'] = "TIME REMAINING"
   label['text'] = count
   if count < 0:
      label['text'] = "Times Up"
      destroy()
   else:
      root.after(1000, countdown, count-1)
root = tk.Tk()
Title = tk.Label(root)
label = tk.Label(root)
Title.place(x=40, y=5)
label.place(x=100, y=30)
countdown(300)
root.mainloop()
```

Fig. 5. *Python Code for Tkinter*

It shows the contestants a timer that they can refer to. After the completion of this timer, the first task above for finding the public/private key has ended and the next phase of the competition starts. In Fig. 6, the task is a bash script to check if the Python script is still running its task. This script will fire off at the end of the time limit to start the following task as described below. Because of the nature of the destroy feature in Tkinter, the program exits before giving its final message. This ensures the timer is kept and the system waits for the next task to start. It fires the Java program and begins the next task.

```
DATE=`date`
OUTPUT=$(ps aux | grep gui.py | grep -v grep)
if [ "${#OUTPUT}" -le 0 ];
then
   echo "Your Time is up"
   echo "New Task will start in 5 seconds"
   sleep 5
   echo -en "\ec"
   java -cp "/home/pi/Desktop/"
fi
```

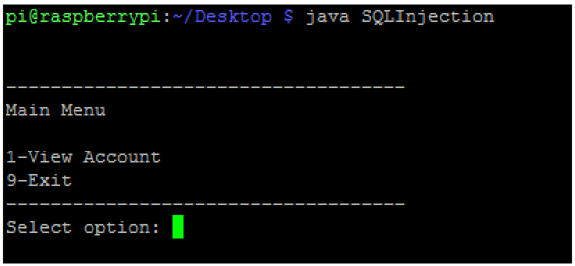
Fig. 6. *Bash Script to Start Next Task*

The second task is to test the ability of an individual to exploit a weakness in a Java program that does not use prepared statements. Instead, the Java program shown in Fig. 7 creates dynamic Structured Query Language (SQL) that directly accesses a MySQL database. By understanding SQL statements and how the query is structured, the contestants need to create an account and give identification, name and balance in the fake bank. This contest is used as a learning mechanism on the proper validation and sanitization of data.

```
private static Account getAccount(String id) {
   Account account = null;
   try {
      Connection conn = getConnection();
      Statement stmt = conn.createStatement();
      String strStmt = "select id, name, balance from
account where id = " + id;
      stmt.execute(strStmt);
      ResultSet rs = stmt.getResultSet();
      if (rs.next()) {
         account = new Account(rs.getInt("id"),
rs.getString("name"), rs.getInt("balance"));
      }
   } catch (Exception e) {
      System.out.println("Error getting the account:
" + e.getMessage());
   }
   return account;
}
```

Fig. 7. *Java Program to Exploit Weakness*

In Fig. 8, the contestants are given a simulated command prompt to enter their proposed SQL injection [9]. The SQL injection is a code injection technique, used to attack data-driven applications, in which nefarious SQL statements are inserted into an entry field for execution. The contestants are required to submit a screenshot with their SQL commands to inject into the database as well as a screenshot with the time of when they took the picture showing their account added with a valid name and balance in the account. The contestants are not given access to the Java code but they are given hints about the structure of the SQL statements.



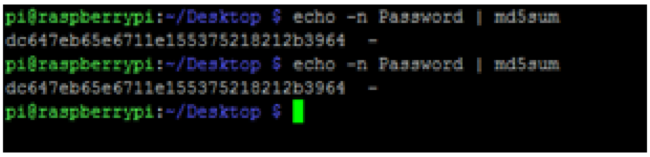
```
pi@raspberrypi:~/Desktop $ java SQLInjection


-----------------------------------
Main Menu

1-View Account
9-Exit
-----------------------------------
Select option: 
```

Fig. 8. *SQL Injection*

The third task includes gaining access to a remote server and setting up a public/private key pair in order to gain permanent access to the server. The server will be communicating with the host of the Raspberry Pi on which the contestants log into using a packet sniffer. The contestants must gain the IP address associated with the server in order to use the Secure Shell (SSH) connection into the server and then set up a public/private key pair or create a backdoor activity using Linux commands. The host will then provide the connection between the server and the clients. In addition, a script will run to overwrite the SSH configuration and to fire for both the host and server. This will disable password authentication on both the host and server machine.

In order to complete the next task, the contestants need to demonstrate that they have accessed to the remote server either via SSH keypairs or through one of the backdoor exploits. This is in preparation for the final phase of the competition. Only contestants that have completed this task may compete for the final phase. This ensures a foundation in basic authentication security for no longer using passwords in dealing with SSH authentication.

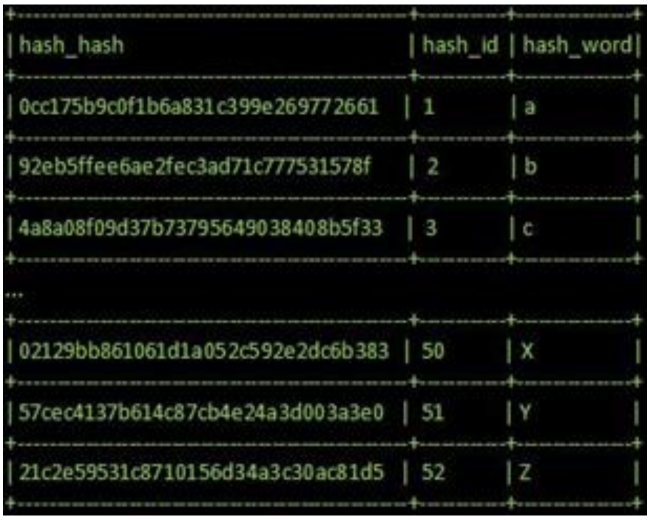
The final task of the competition is to perform a rainbow table [10] attack against an MD5 hash shown in Fig. 9. The MD5 algorithm [11] is a widely used hash function producing a 128-bit hash value. The MD5 hash is given and the contestants need to use this hash to reverse possible passwords. Certain Linux commands run through penetration test suites have this capability as well as online dictionaries. The theory here is to always salt with the hash and use stronger hashing mechanisms. The MD5 hash is an older algorithm that has possible collisions of frames that can reveal patterns in how it hashes passwords. In cryptography, salting is the generation of random data that combines with the hash to create a more unique password that if compromised would not compromise users with similar passwords.



```
pi@raspberrypi:~/Desktop $ echo -n Password | md5sum
dc647eb65e6711e155375218212b3964  -
pi@raspberrypi:~/Desktop $ echo -n Password | md5sum
dc647eb65e6711e155375218212b3964  -
pi@raspberrypi:~/Desktop $ 
```

Fig. 9. *MD5 Hash*

For this task, full access to the server with root permissions is given as this assumes that the contestants have compromised a root account. Though in theory, MD5 hash code is not reversible but using rainbow table databases the contestants can find out which text resulted in a particular hash value. This makes password cracking possible with either brute forcing the crack, for example, putting in passwords until the hashes match or using a pre-generated rainbow table shown in Fig. 10 to speed up the process for known length of passwords.



```
+----------------------------------+---------+-----------+
| hash_hash                        | hash_id | hash_word |
+----------------------------------+---------+-----------+
| 0cc175b9c0f1b6a831c399e269772661 | 1       | a         |
+----------------------------------+---------+-----------+
| 92eb5ffee6ae2fec3ad71c777531578f | 2       | b         |
+----------------------------------+---------+-----------+
| 4a8a08f09d37b73795649038408b5f33 | 3       | c         |
+----------------------------------+---------+-----------+
...
+----------------------------------+---------+-----------+
| 02129bb861061d1a052c592e2dc6b383 | 50      | X         |
+----------------------------------+---------+-----------+
| 57cec4137b614c87cb4e24a3d003a3e0 | 51      | Y         |
+----------------------------------+---------+-----------+
| 21c2e59531c8710156d34a3c30ac81d5 | 52      | Z         |
+----------------------------------+---------+-----------+
```

Fig. 10. *Rainbow Table*

IV. EVALUATION

This project is tested by mimicking one of the competitive teaching methods. This Capture-the-Flag (CTF) application shows how hacking truly works. This project works on a smaller scale in comparison to the extremely complex abilities that the people who participate in these hacking competitions possess [12]. However, it does include the same teaching principles. Up to this day, no one in our university has attempted to address the dangers related to network security. In our project, we manage and create awareness to contestants in an entertaining and educational way to learn hands-on exercises of the fundamentals of network security.

This project helps enforce the core concepts of security and implements a sense of fun when discovering information. By emphasizing the lack of security, we show individuals should be more prudent about what they share and who they share it with. Traffic that goes through the network is easily tracked and readable with many tools. Hacking becomes increasingly easier as technology advances but the way in which network communication occurs remains the same [13].

To build upon our iteration for future works would include an additional router and switch with multiple hosts that would be completely isolated on the layer two network. The programmatic solution to include more advanced topics to challenge those well-versed in security practices. Some advances could include decryption of known weak algorithms and wireless protocols that access via Wi-Fi Protected Setup (WPS) or Wired Equivalent Privacy (WEP) security protocols which are very weakly secured. The attack/defense models of networking security can give a more competitive edge in understanding the more complex topics and how to defend against the automation of network attacks [14].

A precursor to this event could set up to introduce topics included in implementing networks and setting up servers and applications. This would lend itself well to why these attacks work so well on unsecure protocols and why security is a necessity on a more granular level. Helping reinforce these habits will make security concerns come to the forefront of daily interactions with users and data that should be assumed as dangerous. This will make the integration of security a little more seamless and the awareness of the proper balance between ease of use and security.

V. Conclusion

This research exposes programmers in awareness of network security. As a matter of fact, most computer programmers responsible for developing software applications are not aware of the concerns and implications surrounding basic security principles and ideas. We have developed a series of controlled events to interrelate with network security. In our small version of an actual hacking competition, programmers can gain information using built-in features of network communications. In the hands-on application, programmers can gain knowledge about the fundamentals of network security and protect themselves from potential hacker vulnerabilities.

References

[1] Norton by Symantec, "2012 Norton Cybercrime Report", DOI: http://now-static.norton.com/now/en/pu/images/Promotions/2012/cybercrimeReport/2012_Norton_Cybercrime_Report_Master_FINAL_050912.pdf

[2] B. Pashel, "Teaching Students to Hack: Ethical Implications in Teaching Students to Hack at the University Level," Proceedings of the 3rd Annual Conference on Information Security Curriculum Development (InfoSecCD '06), pp. 197-200, September 2006.

[3] D. Bourgeois, "Information Systems for Business and Beyond," Saylor Academy, February 2014.

[4] P. Denning, "Great principles of computing," Communications of the ACM, vol. 46(11), pp. 15-20, November 2003.

[5] M. Mink and F. Freiling, "Is Attack Better Than Defense? Teaching Security the Right Way," Proceedings of the 3rd Annual Conference on Information Security Curriculum Development (InfoSecCD '06), pp. 44-48, September 2006.

[6] J. Hill, C. Carver, J. Humphries, and U. Pooch, "Using an Isolated Network Laboratory to Teach Advanced Networks and Security," Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education (SIGCSE '01), pp. 36-40, March 2001.

[7] J. Harris, "Maintaining Ethical Standards for a Computer Security Curriculum," Proceedings of the 1st Annual Conference on Information Security Curriculum Development (InfoSecCD '04), pp.46-48, October 2004.

[8] A. Kyaw, P. Agrawal, and B. Cusack, "Wi-Pi: a study of WLAN security in Auckland CBD," Proceedings of the Australasian Computer Sceinece Week Mulitconference (ACSW '16), Article No. 42, February 2016.

[9] A. Yeole and B. Meshram, "Analysis of Different Technique for Detection of SQL Injection," Proceedings of the International Conference & Workshop on Emerging Trends in Technology (ICWET '11), pp.963-966, February 2011.

[10] G. Avoine and X. Carpent, "Heterogeneous Rainbow Table Widths Provide Faster Cryptanalyses," Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (ASIA CCS '17), pp. 815-822, April 2017.

[11] J. Touch, "Performance Analysis of MD5," Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '95), pp. 77-86, Augutst 1995.

[12] X. Trabelsi, "Hands-on Lab Exercises Implementation of DoS and MiM Attacks using ARP Cache Poisoning," Proceedings of the 2011 Infromation Security Cirriculum Development Conference (InforSecCd '11), pp. 74-83, September 2011.

[13] V. Vallivaara, M. Sailio, and K. Halunen, "Detecting Man-in-the-Middle Attacks on Non-Mobile Systems," Proceedings of the 4th ACM Conference on Data and Application Security and Privacy (CODASPY '14), pp. 131-134, March 2014.

[14] I. Arce and G. McGraw, "Guest Editors' Introduction: Why Attacking Systsems Is a Good Idea," IEEE Security & Privacy, vol. 2(4): pp. 17-19, October 2004.