On Tackling Social Engineering Web Phishing Attacks Utilizing Software Defined Networks (SDN) Approach

Mohammad Masoud, Yousef Jaradat, Amal Q. Ahmad
Computer and Communication Engineering Department
Al-Zaytoonah University of Jordan, Amman, Jordan
{m.zakaria, y.jaradat, amal.ahmad}@zuj.edu.jo

***Abstract*—Web phishing attacks are one of the challenging security threats. Phishing depends on humans' behavior but not protocols and devices vulnerabilities. In this work, software defined networking (SDN) will be tailored to tackle phishing attacks. In SDN, network devices forward received packets to a central point 'controller' that makes decision on behalf of them. This approach allows more control and management over network devices and protocol. In this work, we propose a neural network based phishing prevention algorithm (PPA) that is implemented utilizing *Ryu*, an open source, SDN controller. The PPA algorithm has been tested in a home network that is constructed with HP2920-24G switch. Moreover, a phished version of *Facebook*, *Yahoo* and *Hotmail* login pages have been written and hosted on three different free hosting domains. PPA has detected all of the phished versions and allowed the access to real version of these services.**

Keywords—Software Defined Networks (SDN); Ryu open source controller; Social Engineering; Phishing; Phishing Preventing Algorithm (PPA).

I. INTRODUCTION

Duo to the Cambrian data explosion era that we live in, millions of subscribers login to online social media accounts, banks accounts, e-mails or companies accounts every day. Many users may use the same credential, username and password, to login to multi Internet services and sites. In 2012, CSID has reported that 61% of American Internet subscribers reuse the same password among multi-websites [1]. In 2013, it has been reported that 55% of Internet users use the same password for almost everything [2]. Moreover, it has been shown by BitDefender that 75% of social sites' subscribers reuse the same username and passwords of their e-mail accounts [3]. These facts demonstrate that if the credentials that a user types in an Internet website is breached, critical and various information of this user in different websites may be compromised. This reduces the efforts and time that hackers need to spend to obtain subscribers critical data. The question is whether is it possible for a hacker to obtain these credentials?

Internet attacks can be categorized into two different classes; service-based and user-based. Service based attacks are defined as the process of attacking the service providers. The attackers study service providers' networks. Subsequently, they exploit its weaknesses. Users and subscribers can regularly change their passwords to prevent against these attacks. On the other hand, User based attacks exploit users' weaknesses even by hacking their PCs, e-mails or by utilizing social engineering. Hacking PCs and e-mails require a deep knowledge in computer programming, operating systems and network protocols. However, social engineering is much easier to perform. Phishing is one example of social engineering attacks that is defined as the process of obtaining critical information from a user by pretending to be trustworthy third party [4].

Phishing is a security challenge because it depends on humans or subscribers' behavior more than on security vulnerability [5]. Phishing can be used in identity theft [6]. It has been reported that 72% of users fall into social phishing [5]. The most popular social phishing example is e-mail phishing. In this type, a fraud e-mail is received from a user asking him to update his information "username, passwords, etc" by clicking a link embedded in the e-mail. When the user clicks the link, it redirects him/her to an identical copy of the real website that he/she subscribes in 'bank account, social website, e-mail account'. When the user complete the provided form in the fraud website copy the information is send to the attacker. Subsequently, the attacker can use the harvested information in different ways.

In this work, we attempt to tackle phishing social engineering attack utilizing software defined networking (SDN). In SDN, a controller is responsible of making decisions rather than network devices. Switches forward received packets to a network controller. The controller directs the switch to handle the received packet in a certain way according to some rule. The controller and switch devices are connected through *Openflow* protocol [7]. Controllers have the ability to inspect the content of packets 'data' and manipulate them. We have expanded this feature to inspect phished links. SDN enabled HP 2920-24G switch has been utilized in this work. *Ryu*, an open source controller, has been optimized and programmed to prevent phishing links. To this end, we propose a phishing prevention algorithm (PPA) that has been programmed and implemented in *Ryu* controller. PPA algorithm is an artificial neural network algorithm that detects if the received webpage is a phished site or not. To examine

the proposed algorithm, a phished version of *Facebook*, *Yahoo* and *Hotmail* login pages have been programmed and hosted on three different free hosting domains. PPA has detected all of the phished versions.

The rest of this paper is organized as follows: Section II overviews some of the related works that have been conducted in phishing detection and prevention field. Section III introduces PPA algorithm. Section V demonstrates the conducted experiment and the obtained results. Finally, section VI concludes this paper.

II. RELATED WORK

This section is divided into two parts. In the first part, phishing detection techniques and related works that have been conducted in this area are presented. The second part will introduce security implementation in SDN paradigm.

A. Phishing Detection Techniques

As mentioned earlier phishing is a part of social engineering attacks. In this attack, a clone webpage of input form is created. Subsequently, this form has to reach the victim through different methods, such as, mobile applications [8], e-mails [9] and short text messages (SMS) [10]. However, the most popular method is e-mail phishing [9]. In this method, attacker sends the link of the clone fraud copy of the webpage to the victim through e-mail. The e-mail contains a message that enforces the user to click on the link and complete the form. After submitting the form, attacker will handle the data.

Many methods and works have been proposed to detect phishing e-mail [11], [12]. In these methods, the authors proposed an approach to detect the e-mail with the fraud links. Features are gathered and collected from e-mails and processed in machine learning approaches. However, these methods cannot work in real time. Moreover, if the link reaches the user through different ways these methods cannot be used. In our work, the clicked links will be investigated. In this way it does not matter how it reaches the victim.

The most relevant work to our work is [13]. The author attempted to find features that can be utilized to detect phished links. However, in his work the author utilized features that cannot be used in a real time. Moreover, many other features from the HTML code can be used. Nevertheless, some of these features have been utilized in this work. Finally, in this work, the proposed approach utilizes the collected features to make a decision of what to do with a link before the user submit its data.

B. Security applications of SDN

Before introducing SDN applications, we will start by overviewing SDN paradigm. SDN is a kind of network where all of network switches act as clients to an intelligent server 'controller'. Switches in this work refer to layer two switches. The switches in SDN network forward any received packet to the controller and then receive instructions for further actions on the packet. The protocol that connects a switch with the controller is the *openflow* protocol. Whenever the controller receives a packet, it executes its programmed algorithm to generate an action that will be sent to the switch. The switch utilizes this action to construct its flow table to generate a consistent action for future packets in the same flow. Figure 1 shows the components of SDN approach

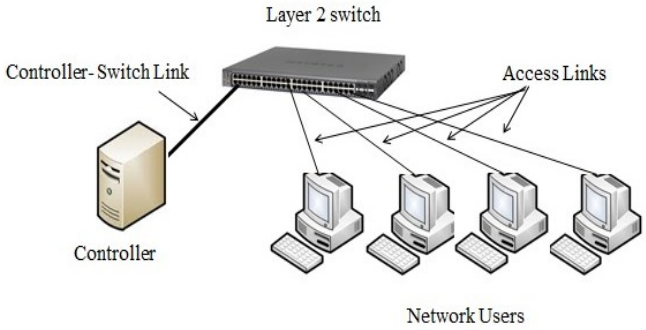

Figure 1: SDN Topology

SDN controller emerged with many flavors. Each one utilizes different programming language and different techniques to be configured. Moreover, there are open and closed sources controllers. In this work *Ryu* controller has been utilized for different reasons. First, it is well documented. Second, it utilizes Python programming language which is easy to follow and learn. Finally, it is an open source controller [14].

SDN approach has been proposed to enhance the administration and management process in networking. This enhancement includes security. Many techniques have been proposed SDN to reduce power consumption [15], managing cloud services [16], implementing Firewalls [17], tackling ARP poisoning Attack [18]. However, for the best of our knowledge we are the first to utilize it to tackle phishing attacks.

III. PROPOSED METHOD

The system consists of two parts. The first part is the proposed PPA algorithm, feature selection, neural network configuration and implementation. The second part is the deployment steps in SDN and *Ryu* extension.

A. Phishing Prevention Algorithm (PPA)

PPA is a real time back propagation neural network based algorithm. The algorithm is a mathematical model with 10 inputs and one output. Each one of these input represents a feature that must be extracted from the received webpages. When these features extracted from webpages, they are used as input data to PPA algorithm. The algorithm output will be a percentage value. Higher value means a phishing page. The utilized percentage value is configured as required from network administrator. In our work we used 50% percentage.

The features used in this work have been selected according to two properties. First they have to be easy to extract from a received webpage with no additional work. Second they have to be simple with no complex structure. We have utilized few features from the work in [13]. In addition we proposed other features. The following are the features that have been used in PPA.

- URL Length: Very long URLs are not good and tiny URL are also not good
- IP address in the URL
- Special characters in URL, such as, '@' or '-'
- Using another '//' in URL
- Number of domains in the webpage
- Mail() function in the form or Mailto()
- HTML header contains any words of the URL
- Containing HTML form
- Port numbers
- Using HTTP or HTTPs

Many Features in [13] cannot be used in real-time implementation since they requires more work, such as, DNS record age, Page Ranked and Number of pages point to the page. Moreover, the content of HTML header and URL has been proposed in this work. We observed that public e-mail services and social networks include the name of their URL in the header and title of HTML code. This motivated us to use this as a feature in PPA algorithm. Finally, the containing form feature has been used since any webpage will be checked and investigated in our system. This feature will accelerate the decision from SDN controller. If the page has no form to fill, the system does not care if it is real page or not since it will not collect any information.

One thing to be mentioned is that PPA is a neural network based algorithm. This means that it as any other neural network supervised algorithms requires training before it can be utilized. The training process of PPA takes place offline before implementing it in SDN network. After the implementation, PPA does not change. In other words, we did not implement a retraining step for PPA after its implementation. The configuration of the back propagation neural network utilized in this work is summarized in table 1.

Table 1: Configuration Parameters

Variable	Value
Number of Hidden Layers	1
Number of Nodes in Hidden Layer	20
Accepted Error Value	0.0001
Training Function	Gradient Descent
Epoch	500

B. PPA-SDN Implementation

After training PPA and obtaining the weights of the back propagation neural network model, these weights have recorded and implemented in a Python function. This function has been implemented in *Ryu* controller configuration file. However, this function cannot be called before extracting its features from the received packets. Each received packet from the switch is opened until reaching the contained data. The content of the packet is used to fill the features required for the implemented controller. The output of this function is used to select one of two commands. The first command is a send action. This action instructs the switch to send the packet. The second action is a drop command. When the switch receives this command, the switch drops the packet. One thing to be mentioned is that the implemented code has been implemented on a simple switch topology to allow *Ryu* to control the switch as a normal layer 2 switch with PPA above it.

SDN approach allowed us to investigate the content of each packet without the need to redirect the traffic or using any kind of man in the middle attack. Figure 2 shows the steps of this algorithm

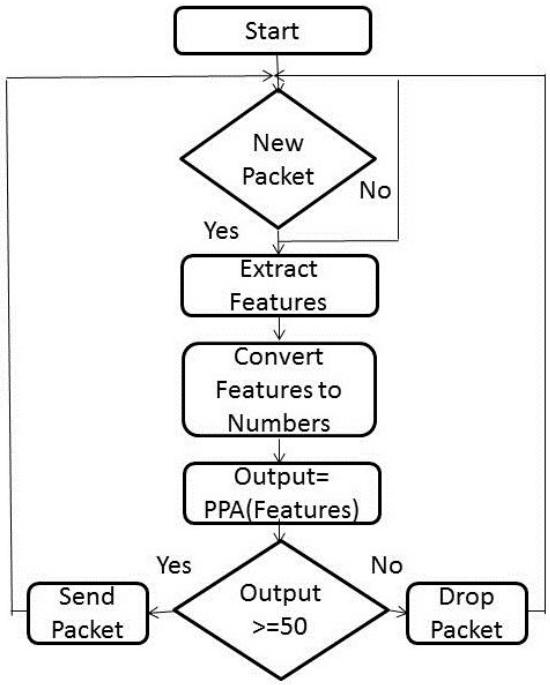

Figure 2: The Proposed System Flowchart

IV. EXPEREMENT

Our experiment consists of three parts. In the first part, training data will be prepared to train and implement PPA algorithm. In the second part, Kali linux will be used to generate the clone webpages that will be used in the phishing attacks. Finally, the SDN topology will be constructed, PPA algorithm will be implemented in *Ryu* controller and the phished webpages will be requested to investigate the accuracy of the proposed model.

A. Data Preparation

Collecting real phishing data is a hard process. Moreover, it requires long time to collect a massive data to train a model. We utilized the data from [13]. However, this data has been prepared with many unrequired features. Moreover, two new features should be included. We have tuned this data to fit our model. More than 11000 records of data have been tuned.

Subsequently, the tuned data have been utilized in MATLAB to train our model. The data have been divided into training, validation and testing with percentage of 60%, 20% and 20%. A regression value of 96.2% has been obtained as shown in Fig.3. Moreover, an MSE value of 0.08 was obtained as shown in Fig. 4. Finally, Fig.5 shows CDF of error value of output data. We can observe that more than 90% of the data have been simulated with zero error.

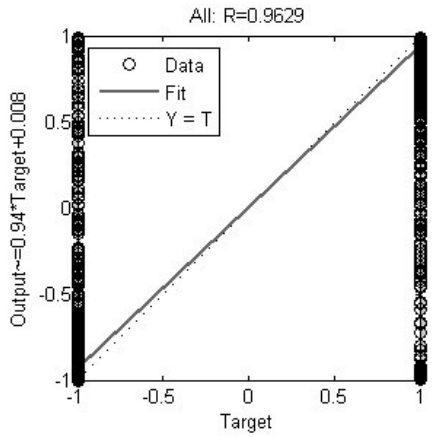

Figure 3: Regression Value

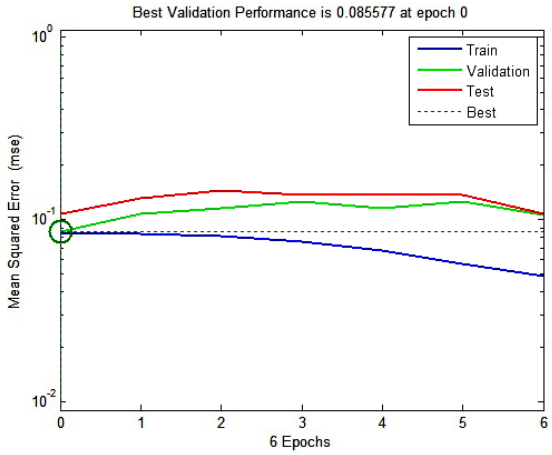

Figure 4: MSE value

After training the model, 241 weights have been collected and used to write PPA model in Python programming language. This function has been added to *Ryu* SDN controller into the simple switch topology. This means that the controller will allow the switch to work as a normal layer 2 switches with a new PPA layer. To reduce delay of the proposed model, we tuned the configuration of *Ryu* to only check the data in frames received from the MAC address of network gateway.

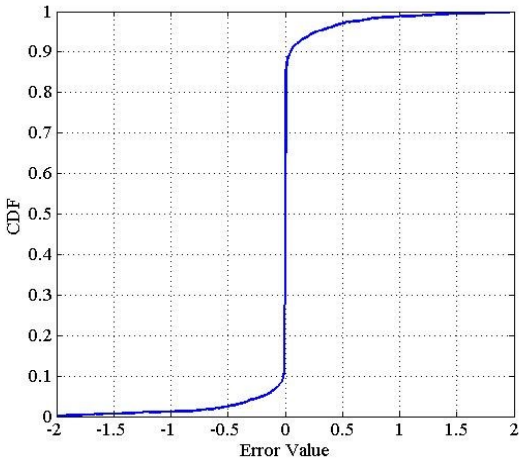

Figure 5: CDF of Error Value

B. Phishing Webpages in Kali Linux

To generate clone websites, first we have to select source websites. Three sites have been selected; *Facebook*, *Yahoo* mail and *Hotmail*. Kali Linux has been utilized to generate these clone sites through social engineering toolkit. This toolkit simplifies the cloning process in a way that the users are not required to write a single line of codes in HTML or CSS. Second, the clone sites must be hosted. Two different hosting styles have been implemented. First a local server with IIS web server installed and a dynamic DNS service configured. Second a free hosting on the web. The generated clones have been taken down after finishing this experiment. Moreover, we pointed the collected data 'if any' from these clone to null address. Finally, URL shortening services have been utilized in the final stage of the experiment to investigate their impact on PPA. *Goo.gl* service has been utilized to cover the IP address of the local server with a new link. It also has been used to shortening the URL of the free hosting websites.

C. SDN Implementation

Fig.6 shows the experiment environment. *Ryu* SDN Controller has been downloaded and installed on Ubuntu Linux. Two different hosts have been implemented. The first host has Windows 7 operating system. The second host has Kali Linux. Kali Linux host has been used to generate the clone fake webpages. We also used it in our experiment as a host that will request these pages. A forth with Windows server and IIS web server installed and configured to serve our sites. HP 2920-24G switch is used as SDN switch. One thing to be mentioned is that this switch can be replaced with any computer or a Raspberry Pi kit with vSwitch open source software SDN switch installed. Moreover, the controller also can be embedded in the same machine to reduce the utilized equipment in home configuration. However, if this solution is to be implemented in offices, SDN switch should be implemented to enhance performance.

The conducted experiment is divided into four scenarios. In the first one, the clone websites have been hosted in our server. The URLs of these pages were the global IP address of our server. In the second scenario, a free dynamic DNS service has

been configured. With a new domain name of each cloned site. A good thing about dynamic DNS service is that we kept our sites hosted on our server machine.

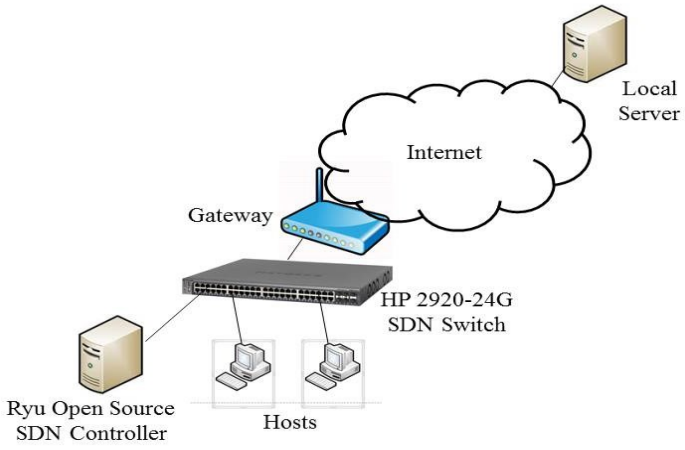

Figure 6: Experiment Environment

The third conducted configuration is to host the sites on free online hosting service. Finally, the last configuration was shortening the hosted sites' URL with an online shortening service. Our two hosts requested the pages in these four scenarios with an additional scenario of requesting the real pages from their legitimate URLs. Table 2 shows the output of these scenarios.

Table 2: Results of the Five Scenarios

Scenario	Windows Host	Linux Host
legitimate URLs	Access	Access
IP URL	Denied	Denied
Dynamic DNS Hosting	Denied	Denied
Free Online Hosting	Denied	Denied
Shortening URL Servie	Denied	Denied

V. CONCLUSION

Social Engineering web phishing is a security challenge since it depends on human behaviors more than protocol or devices vulnerabilities. These kinds of attacks can be utilized without vast knowledge in computer programing or network protocols. In this work, Phishing Prevention Algorithm (PPA) was proposed. PPA algorithm utilizes back propagation neural network model. To implement this algorithm, SDN approach has been adopted with *Ryu* controller. An experiment has been designed and conducted with five different scenarios to examine the accuracy of the proposed model. SDN approach with PPA algorithm running on its controller was able to detect and deny the access to phished sites. In the future we will attempt to gather more online phished sites using PPA implementation.

ACKNOWLEDGMENT

This research was supported in part by Al-Zaytoonah University of Jordan fund (2/11/2014). We would like to thank the University for the equipments and tools they provided for this work

REFERENCES

[1] CSID, "CONSUMER SURVEY: PASSWORD HABITS A study of password habits among American consumers", September 2012, [online] https://www.csid.com/wp-content/uploads/2012/09/CS_PasswordSurvey_FullReport_FINAL.pdf

[2] OfCom, ""Adults' Media Use and Attitudes Report", 2013 , [onlinet] http://media.ofcom.org.uk/news/2013/uk-adults-taking-online-password-security-risks/

[3] SecurityWeek, http://www.securityweek.com/study-reveals-75-percent-individuals-use-same-password-social-networking-and-email

[4] Jagatic, Tom N., et al. "Social phishing." *Communications of the ACM* 50.10 (2007): 94-100.

[5] Kathirvalavakumar, Thangairulappan, Krishnasamy Kavitha, and Rathinasamy Palaniappan. "Efficient Harmful Email Identification Using Neural Network." *British Journal of Mathematics & Computer Science* 7.1 (2015): 58.

[6] Drake, Christine E., Jonathan J. Oliver, and Eugene J. Koontz. "Anatomy of a Phishing Email." *CEAS*. 2004.

[7] McKeown, Nick, et al. "OpenFlow: enabling innovation in campus networks." *ACM SIGCOMM Computer Communication Review* 38.2 (2008): 69-74.

[8] Marforio, Claudio, et al. "Personalized security indicators to detect application phishing attacks in mobile platforms." *arXiv preprint arXiv:1502.06824* (2015).

[9] Chandrasekaran, Madhusudhanan, Krishnan Narayanan, and Shambhu Upadhyaya. "Phishing email detection based on structural properties." *NYS Cyber Security Conference*. 2006.

[10] Saxena, Neetesh, and Ashish Payal. "Enhancing Security System of Short Message Service for M-Commerce in GSM." *International Journal of Computer Science & Engineering Technology (IJCSET)* 2.4 (2011).

[11] Drake, Christine E., Jonathan J. Oliver, and Eugene J. Koontz. "Anatomy of a Phishing Email." *CEAS*. 2004.

[12] Fette, Ian, Norman Sadeh, and Anthony Tomasic. "Learning to detect phishing emails." Proceedings of the 16th international conference on World Wide Web. ACM, 2007.

[13] Mohammad, Rami, McCluskey, T.L. and Thabtah, Fadi Abdeljaber (2014) Intelligent Rule based Phishing Websites Classification. IET Information Security, 8 (3). pp. 153-160. ISSN 1751-8709

[14] Monaco, Matthew, Oliver Michel, and Eric Keller. "Applying operating system principles to SDN controller design." *Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks*. ACM, 2013.

[15] Kannan, Kalapriya, and Subhasis Banerjee. "Compact TCAM: Flow entry compaction in TCAM for power aware SDN." *International Conference on Distributed Computing and Networking*. Springer Berlin Heidelberg, 2013.

[16] Banikazemi, Mohammad, et al. "Meridian: an SDN platform for cloud network services." *IEEE Communications Magazine* 51.2 (2013): 120-127.

[17] Hu, Hongxin, et al. "Towards a reliable SDN firewall." *Presented as part of the Open Networking Summit 2014 (ONS 2014)*. 2014.

[18] Masoud, Mohammad Z., Yousf Jaradat, and Ismael Jannoud. "On preventing ARP poisoning attack utilizing Software Defined Network (SDN) paradigm." *Applied Electrical Engineering and Computing Technologies (AEECT), 2015 IEEE Jordan Conference on*. IEEE, 2015.