Phishing-Aware: A Neuro-Fuzzy Approach for Anti-Phishing on Fog Networks

Chuan Pham*[§], Luong A. T. Nguyen[‡], Nguyen H. Tran*[¶], Eui-Nam Huh*, Choong Seon Hong*,
*Department of Computer Science and Engineering, Kyung Hee University, Korea,
[§]Synchromedia - École de Technologie Suprieure, Université du Québec, H3C1K3, Canada,
[¶] School of Information Technologies, The University of Sydney, NSW, Australia.
[‡]Department of Computer Science, HCMC University of Transport, Viet Nam.

***Abstract*—Phishing detection is recognized as a criminal issue of Internet security. By deploying a gateway anti-phishing in the networks, these current hardware-based approaches provide an additional layer of defense against phishing attacks. However, such hardware devices are expensive and inefficient in operation due to the diversity of phishing attacks. With promising technologies of virtualization in fog networks, an anti-phishing gateway can be implemented as software at the edge of the network and embedded robust machine learning techniques for phishing detection. In this paper, we use uniform resource locator (URL) features and web traffic features to detect phishing websites based on a designed neuro-fuzzy framework (dubbed Fi-NFN). Based on the new approach, fog computing as encouraged by Cisco, we design an anti-phishing model to transparently monitor and protect fog users from phishing attacks. The experiment results of our proposed approach, based on a large-scale dataset collected from real phishing cases, have shown that our system can effectively prevent phishing attacks and improve the security of the network.**

***Index Terms*—Phishing websites, Neuro-fuzzy network, Neural network, Fuzzy, Fog computing, Cloud computing.**

I. Introduction

A. Phishing websites

Phishing is a criminal activity that steals victims' personal information using misleading emails or fake websites [1]. The word "phishing" is originated from the word "fishing" [2]. Online users can be easily deceived into entering their personal information because phishing websites are highly similar to real ones. Maliciously, by creating phishing sites, "phishers" use a number of techniques to fool their victims, including email messages, instant messages, forum posts, phone calls, and social networking information [3]. Phishing results in severe economic loss all over the world, and phishing sites are also growing rapidly in quantity and complexity. According to reports from the Anti-Phishing Working Group [3], the number of phishing attacks is increasing by 5% monthly. Fig. 1 illustrates the urgency and importance of phishing identification in modern society, which is based on a phishing website report received in the first quarter of 2016 [3].

However, at the edge of networks, the anti-phishing problem has not been well-addressed due to the following reasons. First, mobile users check their emails and use web browsers more frequently than desktop users [4]. Thus, they are much more likely to access on phishing sites that have not yet been detected or taken down by anti-phishing applications and firewalls at their local networks or on their devices. Second, mobile devices are always "hungry" for energy and computing resources (e.g., limitations of CPU, memory, and user interfaces), so anti-phishing tools are usually ignored or removed on these devices. Hence, it is hard for users to discern if an incoming link is legitimate or not. Third, existing anti-phishing tools (e.g., default plug-ins on web browsers or local anti-phishing applications) are inefficient in terms of detection (this will be analyzed concretely later in Section III), and mobile users may be exposed to phishing attacks when engaging in usual behaviors. According to the report [5], mobile users are three times more likely to submit their login information than desktop users do. Therefore, preventing phishing attacks against terminal users is a critical issue in the edge of networks.

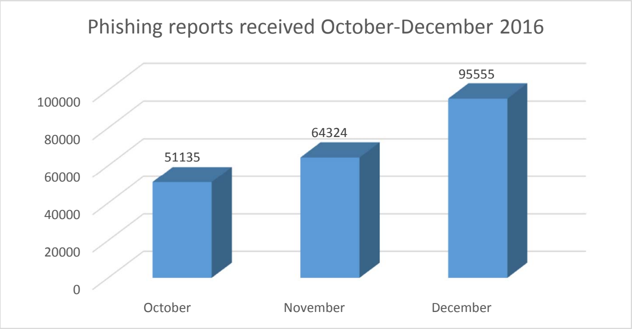


Fig. 1: Phishing reports received in the period of October-December 2016 [3].

As discussed in [6], there are three classes of technical methods to identify phishing websites, including the blacklist/whitelist methods [7], [8], the web structure-based methods [9] and the web content-based methods [6]. The blacklist methods are often deployed in practice due to their inexpensive cost and speed of detection [10]. On the other hand, although web content-based methods can detect phishing websites with high accuracy [6], they are difficult to apply in real-time detection. The network operator can combine content-based

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the Grand Information Technology Research Center support program (IITP-2018-2015-0-00742) supervised by the IITP (Institute for Information and communications Technology Promotion).
Corresponding author: Dr. C.S. Hong (Email: cshong@khu.ac.kr).

methods and blacklist/whitelist methods by regularly creating a large amount of automated agents to collect webpages or receive phishing reports from users, then analyze the content and update the blacklist/whitelist database [11]. In this paper, we design a neuro-fuzzy network model that uses detection features of the blacklist/whitelist and web-structure methods. Our approach improves not only the accuracy of phishing identification compared to existing methods, but also the detection performance at the edge of the network.

B. Fog computing paradigm and anti-phishing services on fog

Focusing on the edge of networks, Cisco recently introduces the concept of fog computing, which extends the cloud so that it is closer to users [12]. Fog computing also provides data, computing, storage and application services to end-users as well as cloud computing. In addition, at the edge of networks, fog supports high mobility and a dense geographic distribution [12]. Strong characteristics of fog networks now pull services provided at places near the end-users, such as access gateways, or even the set-top-box, as shown in Fig. 2.

Can we use the advantages of fog networks to develop anti-phishing tools at the edge of networks, where a base station (BS) or an access point (AP) can support security services for mobile devices? It is possible with the help of network function virtualization (NFV) [13] in the fog network, where a fog node now has sufficient resource capacity to virtualize any network function, such as a firewall, an anti-virus, and an anti-phishing function. Further, it is also more tractable and significant to embed machine learning techniques on fog nodes than on hardware-based devices. A fog-based service can enhance the performance compared to existing cloud-based methods [14]–[16] since fog nodes are located at the edge of networks. Last but not least, deployment of an anti-phishing service on fog nodes does not degrade the computing resources of mobile devices as much compared to installing directly anti-phishing applications on these devices.

For the client side, by adding an anti-phishing gateway at the network edge, phishing websites can be stopped before reaching user devices, so an identification procedure can work transparently for terminal users. In this paper, we design a neuro-fuzzy model in the fog network (called Fi-NFN) to detect phishing URLs in real time. We further mitigate the arbitrary *IF-THEN* rule set in the phishing detection of the neural network model of [17] and combine the neural network with the fuzzy model to improve the performance. The main contributions of this paper are summarized as follows:

- We design input features, which combine by three useful URL features (PrimaryDomain, SubDomain, PathDomain) and three web traffic features (PageRank, AlexaReputation and GoogleIndex). Especially, we use GoogleIndex instead of using Google search results like traditional methods to improve the accuracy (this will be explained later in Section III). These features are more readily available and faster than web-content features [18] in terms of gathering features and detecting phishing websites.
- We develop a new neuro-fuzzy approach without using *IF-THEN* rules to identify phishing. We are motivated by a prior study [17] that used the conventional neural network model. Combining the neural network with the fuzzy model, we obtain a good result in terms of identification accuracy.
- Our Fi-NFN classification model enhances the classification accuracy to 98.36% and improve the convergence of the training phase. Furthermore, our system can achieve real-time response and stable performance to detect phishing URLs.
- To apply phishing identification using a neuro-fuzzy method in a fog network, we propose a detection framework to protect the terminal users. The framework contains with two components: the identification component and the back-end component. The identification component is deployed at the fog nodes to observe and detect URL requests. This component is aware of danger and prevents users from accessing phishing websites. The back-end component is placed on the cloud and plays a role as a management tool. One of the important contributions of our paper is we design a model with two isolated components that are adapted to the fog architecture. This is helpful in detecting phishing sites wih real-time responses at the fog nodes and easily maintaining the system for tasks, such as training and updating parameters for the identification component.
- In order to validate the efficiency of our model, we conduct extensive simulations. The results show that our model outperforms state-of-the-art methods in terms of the accuracy of phishing detection and the response time.

The rest of the paper is organized as follows. In Section II, we discuss related work. In Section III, we describe the background knowledge and propose the system model on the fog network. The neuro-fuzzy approach is introduced and illustrated concretely in Section IV. All simulations and numerical results are discussed in Section V. Finally, we conclude our work in Section VI.

II. Related Work

In this section, we present the information related to state-of-the-art anti-phishing mechanisms and fog computing that is essential to the theory behind our work. There are a lot of different studies regarding phishing identification including phishing websites, phishing emails and phishing applications. Here, we only investigate phishing websites and research trends related to our work.

A. Anti-phishing mechanisms

As mentioned above, the blacklist/whitelist methods are used in many studies and in practice [7], [8], [15], [16], [19], [20]. However, these methods has to maintain a list of phishing websites using a manual/automatical update process as shown in Fig. 3. URL requests are checked before launching based on a local database or a database on the cloud. Even though the blacklist/whitelist techniques has quick detection, managing the blacklist/whitelist database is inefficient for both the local database and the cloud database due to the rapidly increasing number of phishing sites. Therefore, heuristic and machine

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2831197, IEEE Transactions on Network and Service Management

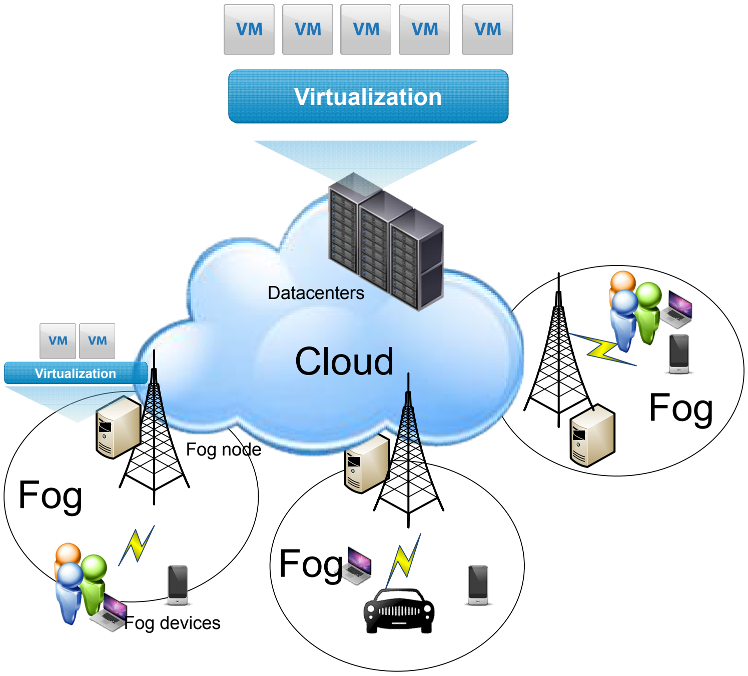


Fig. 2: Fog computing architecture: using virtualization techniques, fog nodes can provide services at the edge of a network.

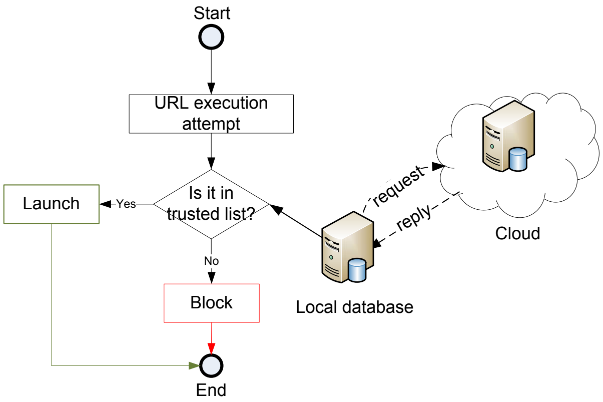


Fig. 3: The blacklist/whitelist mechanism in phishing detection.

learning approaches have been received much attention in terms of automatic detection. This can be seen as a hybrid approach that uses blacklist/whitelist methods to protect users at the front end (i.e., the client side), while machine learning techniques are used at the back end (i.e., the server side) to detect phishing and update on the blacklist/whitelist database.

Several machine learning techniques focus on the web structure or web content-based methods to detect phishing URLs [6], [17], [18], [21]–[26]. One well-known approach in extracting web page features to detect phishing sites is Cantina+ [18]. This approach is based on 15 features web pages, including URL features and content features, which are expensive during the analysis. Another robust work focusing on URL features [26] presented the effectiveness of many approaches in the classification of phishing URLs; however, they did not concretely consider the combination of multiple features to enhance the identification performance.

Based on certain page-ranking features, the authors in [25], [27] developed web structure-based approaches to identify phishing web pages by using the Google PageRank value. However, using only the PageRank value is insufficient to identify phishing URLs due to the following reasons. First, many phishing websites are created on popular websites such as blogs or Google sites, where the ranking features (e.g., domain age [25]) are not useful for phishing identification. Second, new URLs have low ranking values that are similar to phishing URLs. In some specific cases, we cannot obtain PageRank values from the API correctly. Such specific cases cannot be detected by [18] and [25]. Hence, combining many features is necessary to reduce the amount of missed detections for phishing identification.

In other trends of the machine learning approach, phishing websites can be identified based on a rule set [21], [22], [24]. In [24], the authors proposed a fuzzy technique based on 27 features of a web page, classified into three classes. Using a rule set [21], [22], they obtained fast detection. However, there are some weaknesses in their methods. For example, the rule set is not objective and greatly depends on the developer. Another limitation is the weights of each main criteria [24] that are used without clarification. Finally, heuristic parameters are very sensitive and difficult to apply in practice due to the complexity of phishing sites.

Another approach in machine learning is the neural network [17] that has attracted much attention in the literature. However, this work did not clarify the design of input features and the neural network model, such as how to represent input features in the network, and how to integrate the phishing information to train the network. To improve the performance of the neural network model, we design a neuro-fuzzy network model with six URL input features. We can easily integrate the phishing information into the neuro-fuzzy network through the learning process to enhance the convergence of the training phase.

Toward implementing a phishing identification framework in practice, we next discuss the trend of deploying services in networks, where multiple services are pulled to execute at the edge of networks.

B. Fog computing: A practical environment

With the development of cloud computing, there are billions of devices connected to the Internet that require mobility, geo-distribution, location awareness and low latency [28]. Fog computing, a new network paradigm, is an extension of cloud computing and services to the edge of the network. The comparison between cloud computing and fog computing in [29] shows advantages of the fog computing. It becomes a promising architecture to be adapted to Future Internet.

Specifically, prior studies in fog networks [30], [31] have proposed security models, in which a fog node can play a role as a gateway to identify illegal transactions from fog users. For anti-phishing issues, models that are used to protect clients as gateways or firewalls are already implemented, such as [15], [16], [19], [32]; however, they mostly rely on traditional proprietary purpose-built hardware. A fog-based anti-phishing deployment can automatically detect phishing URLs. Furthermore, we can flexibly embed machine learning techniques to improve performance [12] since a fog node has powerful computing resources. In particular, fog nodes can be deployed underlying network function virtualization technologies, in which anti-phishing tools can be run as a virtual machine and share resources with other functions of the fog nodes, such as routers or gateways [33]. Consequently, for-base implementation is our target to deploy an anti-phishing service at the edge of network.

III. Identifying Phishing Sites and The System Model on Fog Networks

To easily understand how to identify phishing sites, in this section, we briefly discuss some background knowledge related to phishing identification and illustrate our proposed anti-phishing model in fog.

A. Preliminaries

1) Phishing identification tools: Following the approach using URL and web traffic features, we show some famous techniques used to detect phishing websites and related to our work.

- **WHOIS.** The WHOIS function provides details about the date of registration, update and expiration, the registrar [34]. Phishing sites are often unstable, and their registration dates are often newer than those of the legitimate sites. Moreover, many phishing sites contain IP addresses in their URLs [35]. Therefore, WHOIS is a helpful tool for detecting phishing sites.
- **DNS Blacklist.** There are many blacklist providers that contain a list of phishing sites. These providers frequently update their database and support query methods for users, for example SORBS [36], URIBL [37], and SURBL [38].
- **Browser toolbars.** Browser toolbars provide a client-side defense for user browsers [23]. Whenever a user visits a website, the browser toolbar will filter URLs from the address bar, then refer to a blacklist database. If the URL exists on that database, a special warning will be responsed to the user. Google Toolbar [20] is a popular tool integrated as a Firefox browser extension. In addition, there are several safe browsing tool bars that work with Chrome, Safari, and Internet Explorer [20].
- **Network Appliance.** To combat phishing and other Internet attacks, a network appliance, such as a firewall or a gateway in a network, can be implemented. Trend Micro [19] and Symantec [39] have developed joint Internet access and security solutions as a safeguard [32]. Such hardware tools often need to refer local or online URL blacklist databases. They can carry out well a small network; however they lack of flexibility in update that needs to adapt to diversity of phishing websites. Furthermore, some middle-layer defense models and third-party models are proposed in practice [40], even though they still raise much discomfort level in mobile users as they require many communication steps to protect mobile users.

In the next part, we illustrate the identification features and discuss how to use them in phishing detection.

2) Identification features: Phishers usually try to make the Internet addresses (URLs) of phishing sites similar to legitimate sites to fool online users. However, they cannot reuse URLs of legitimate sites that are already registered. Based on various characteristics of URLs, we indicate the differences between a legitimate URL and a phishing URL.

- **Features of URL.** The structure of URL is as follows: $< protocol >$: $// < SubDomain > . < PrimaryDomain > . < TLD > / < PathDomain >$. For example, the URL: *http://paypal.abc.net/ index.htm* includes the following six elements: the protocol is *http*, the SubDomain is *paypal*, the PrimaryDomain is *abc*, the top-level domain (TLD) is *net*, the Domain is *abc.net*, and the PathDomain is *index.htm*.

  There exist many differences between phishing URLs and legitimate URLs that can be used to recognize easily based on URL features. In particular, we describe in detail three features: the PrimaryDomain, SubDomain and PathDomain of the URL.

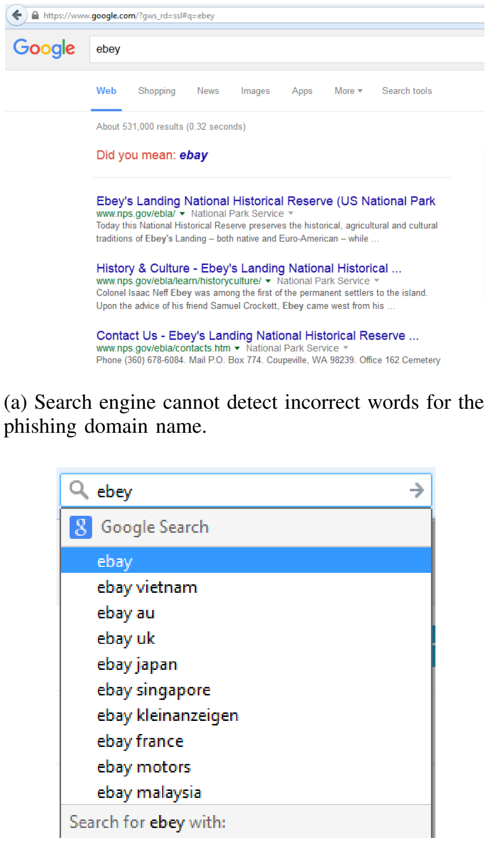
  - PrimaryDomain: Phishers cannot use the original PrimaryDomain since it is already registered by the original company. Hence, phishers register misspellings or similar PrimaryDomain of phishing websites to fool users. For example, URL *www.paypall.com* looks similar to the well-known website *www.paypal.com*.

  - SubDomain: Phishers often prepend the domain of phishing websites to their website. For example, phishers prepend the SubDomain "*paypal.com*" to any other domain (e.g., ".io", ".biz") that may fool users into the phishing URLs.
  - PathDomain: This is a sub-folder of the URL. Phishers can also use the PathDomain to fool users. For example, phishers may navigate users to the URL *www.attack.com/paypal*, where a phishing website interface is similar to the original one. Carelessly, the users will think that this URL is from the "*paypal.com*" site. Especially, using mobile devices with small graphic interfaces, it may be too difficult to recognize such phishing URLs.

  Using these features, we can identify a phishing website by measuring the similarity score between legitimate and phishing websites. However, it is inapplicable to be used only these features in practice since a legitimate website owns multiple similar URLs. For example, although "nld.com.vn" has a high similarity score with the legitimate URL, "nld.vn", it is also a legitimate URL that points to the same website. Multiple similar URLs, used for load balancing techniques, are deployed in many web services, which can be a challenge when using this score. Therefore, we combine URL features with web traffic features to improve the performance.

- **Features of web traffic.** Most of the lifetime, legitimate websites are safe for users to browse. Thus, they have high ranks from search engines [25]. Meanwhile, phishers usually create fake sites to mimic famous sites. Such phishing sites have low ranks. Espcially, phishers can not fake ranking values from search engines and ranking systems [41]. The famous ranking systems are used in our work as follows
  - PageRank [25], [42]: Google search engine uses a link analysis algorithm [43] to build PageRank values. Most phishing web-pages have low PageRank, because these sites exist only for a short time.
  - AlexaReputation [44]: AlexaReputation value of a website is calculated as the number of links from other webpages to itself. AlexaReputation is similar to Pagerank, where AlexaReputation values of phishing websites are much lower than the values of the legitimate sites.
  - GoogleIndex [45]: Google index lists all legitimate sites that are visited by agents of Google. Google frequently updates this index list for its search engine. The values of GoogleIndex for phishing websites are much smaller than those of legitimate sites.

  In this paper, we use famous ranking systems to identify phishing sites. They look similar; however, combining them can improve the accuracy of detection due to the following reason. First, with new URLs that have just been created, GoogleIndex system returns empty values, while others can compensate with positive values. Second, GoogleIndex is not a ranking system, but it owns huge dataset and trusted results. This combination reflects exactly the lifetime of URLs.



About 531,000 results (0.32 seconds)

Did you mean: ***ebay***

Ebey's Landing National Historical Reserve (US National Park
www.nps.gov/ebla/ ▾ National Park Service ▾
Today this National Historical Reserve preserves the historical, agricultural and cultural traditions of **Ebey's** Landing – both native and Euro-American – while ...

History & Culture - Ebey's Landing National Historical ...
www.nps.gov/ebla/learn/historyculture/ ▾ National Park Service ▾
Colonel Isaac Neff **Ebey** was among the first of the permanent settlers to the island. Upon the advice of his friend Samuel Crockett, **Ebey** came west from his ...

Contact Us - Ebey's Landing National Historical Reserve ...
www.nps.gov/ebla/contacts.htm ▾ National Park Service ▾
Phone (360) 678-6084. Mail P.O. Box 774. Coupeville, WA 98239. Office 162 Cemetery

(a) Search engine cannot detect incorrect words for the phishing domain name.

(b) Incorrect words for the phishing domain name do not exist in GoogleIndex.

Fig. 4: Comparison of detecting phishing domain names between the Google search engine and GoogleIndex.

  Other features, such as special characters in URLs or the number of dots, the length of URL, can be used to detect phishing websites [10], but they are really specific, and attackers can replace or fake them easily. In this work, we focus on detecting phishing attacks in real time. Hence, the system has less time to analysis and make a decision. Therefore, we do not select identification features that cannot analyze in real time.

GoogleIndex is a new feature in our work, meanwhile traditional methods are based on the Google search results [17], [18], in which they use the query results from the Google search engine to lookup phishing terms in a domain name or a host name. If the received results are in the top (e.g., top 30), they will not be considered as phishing sites. Unfortunately, based on search engines, many phishing sites cannot be recognized. For example, for a phishing website that changes the domain name from "ebay" to "ebey", Google search engine stills show this term in top 30 as illustrated in Figure. 4(a). Meanwhile, the GoogleIndex does not index this term in the suggestion system, which rarely appears in query history, as shown in Figure. 4(b).

Next, we discuss the background needed for the neuro-fuzzy network in the next part, in which we present clearly all the design layers as well as computations to train and identify

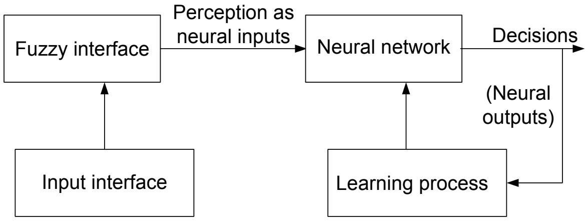


Fig. 5: Flow chart of the neuro-fuzzy model.

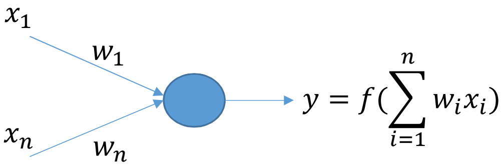


Fig. 6: A regular fuzzy neural network.

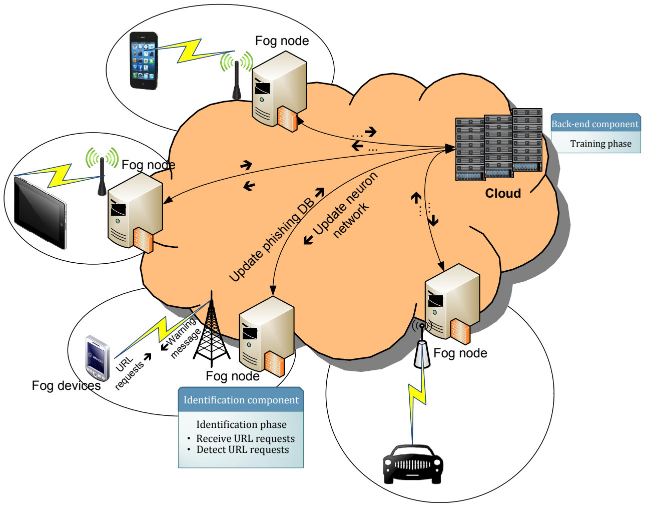


Fig. 7: Phishing identification architecture for the fog network.

phishing URLs.

3) Neuro-fuzzy network system: A neuro-fuzzy network refers to a combination of the artificial neural network and the fuzzy logic in the field of artificial intelligence [46]. For neural networks, the knowledge can be automatically achieved based on the backpropagation training, but the learning process is slow. Also, it is difficult to integrate special information about the knowledge of training datasets to improve the learning process [46]. For fuzzy systems, they are restricted to the fields where the knowledge is available to build the rule set and where the number of inputs is small. To overcome the problem of knowledge acquisition, the cooperative approach, i.e., a neuro-fuzzy network, can optimize certain parameters of fuzzy systems as well as enhance the training process of the neural network. In the literature, the efficiency of the neuro-fuzzy network approach is shown in several approaches, such as image processing, pattern recognition. In this work, we advocate the neuro-fuzzy network approach for phishing identification in fog network.

There are some hybrid models of the neuro-fuzzy network. Here, we design a neuro-fuzzy network following the basic model as depicted in Fig. 5. At first, in response to the input parameters, the fuzzy interface module provides an input vector to the neural network. Via the fuzzy module, specific information for the input data is embedded before training in the neural network, which can improve the training performance and decision process.

Consider a simple neural network, as shown in Fig. 6. We present mathematical calculations relevant to this neuro-fuzzy network as follows. The input signal is a vector $\boldsymbol{x}$ with n elements x_i that interact with the weight vector $\boldsymbol{w}$ to produce a vector $\boldsymbol{p}$ by $p_i = w_i x_i, i = 1, 2, ..., n$. The information is aggregated as follows:

$$\text{net} = \sum\nolimits_{i=1}^{n} p_i. \tag{1}$$

The neuron uses an active function $f(t)$ (e.g., a sigmoid function $f(t) = \frac{1}{1+e^{-t}}$) to compute the output:

$$\text{y} = f(\text{net}) = f(\sum\nolimits_{i=1}^{n} p_i). \tag{2}$$

A fuzzy model can be integrated differently into a neural network model to improve the training phase [47]. Fig. 5 shows a design, where the membership function sigmod [46] is used to fuzzify and defuzzify the input values. Similar to the neural network model, the fuzzy output is now calculated as follows:

$$\text{Y} = f(\text{net}) = f(\sum\nolimits_{i=1}^{n} w_i X_i). \tag{3}$$

where X is the fuzzy input signal and $f(t) = (1 + e^{-t})^{-1}$ is the sigmoid function. By Zadehs extension principle [48], the membership function of the fuzzy output Y is calculated as follow

$$Y(t) = \begin{cases} (\sum_{i=1}^{n} w_i X_i)(f^{-1}(t)) & \text{if } 0 \leq t \leq 1, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where $f^{-1}(t) = \ln t - \ln(1 - t)$.

The above example represents a simple computation in the neuro-fuzzy network. Corresponding to the input parameters and applications, the design of the neuro-fuzzy network may be different. In the next section, we discuss the architecture of the neuro-fuzzy network to detect phishing URLs in the fog networks.

B. Identifying phishing websites in the fog architecture

Fog computing is a potential approach to integrate security applications at the edge of networks [30]. A fog node is close to users and can be a local central node, where all traffic from users is centralized at the fog node. Intrusion detection techniques can be implemented on a fog node to detect user behaviors. In this paper, we implement a detection application on a fog node to identify phishing sites. When local devices (e.g., cloud users and mobile users) send requests to access the Internet, the fog node transparently detects request

URLs, then restricts or notifies users, if request URLs are phishing. In this case, the fog node plays the roles of a monitor and a firewall. It also illustrates that a fog node can protect users without installing anti-phishing tools on the local devices. Last but not least, detecting on the fog nodes is a "quiet" and "smart" process to the end users since this does not require any configuration on their part or raise a series of confusing questions or warnings during detection, as the traditional methods have been done.

In Fig. 7, we design two distinct components in our phishing identifying model: the identification component on a fog node and the back-end component in a data center of the cloud. The identification component is integrated on a fog node and interacts with fog users. It contains a neuro-fuzzy network that is already trained to classify URLs into two classes: the phishing URL class and the legitimate URL class. There is a connection between these components to update trained parameters of the neuro-fuzzy network. On the cloud, the back-end component also has the same neuro-fuzzy network architecture as the component in the fog nodes. This component invokes the training phase in order to update parameters of the neuro-fuzzy network. It then synchronizes all parameters from the cloud component to the fog node component. This step does not spend large network traffic or time consumption to update the phishing database compared to the blacklist method (the network traffic measurement is discussed later in Section V). Further, the training procedure can be invoked and adjusted easily by administrators in the back-end component. Finally, the training phase and updating phase do not impact the identification phase in a fog node. Our anti-phishing model not only reduces the detection latency at fog nodes, but also alleviates the complex computations at the edge of a network. At a fog node, the propagation from input URLs to the output can be executed in real time and transparently with regard to the users.

In summary, our proposed architecture on fog networks can achieve higher performance compared to existing methods due to the following reasons:

- Anti-phishing tasks are executed at fog nodes, which can free up the mobile device's resources from local installations.
- Fog nodes are closer to the user devices than the cloud; thus the response of detection is faster than that of services deployed on the cloud.
- Fog nodes have powerful resources to integrate and execute the phishing identification application in real time based on the neuro-fuzzy network. It is also able to update and train the network without degrading the nodes' resources.

IV. Neuro-fuzzy-based Phishing Identification Model on The Fog Network

A. The five-layer neuro-fuzzy network architecture

In this subsection, we present in detail our model for URL classification. The designed neuro-fuzzy network model (called Fi-NFN) is illustrated in Fig. 8. Fi-NFN includes five layers, combining the fuzzy model and neural network model as follows.

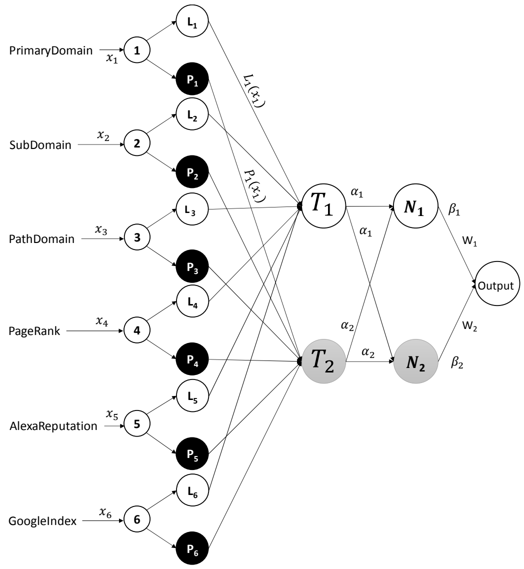


Fig. 8: The five-layer neuro-fuzzy network (Fi-NFN) architecture. This architecture includes six input nodes, five layers and one output node.

- The first layer, called the input layer of Fi-NFN, contains six nodes. The values of those nodes are crisp values (defined by the vector $\boldsymbol{x}$) of the fuzzy module. Each element value of x is extracted from the features described in Section III-A2. Three of six input features are text strings that need to be represented as real values for input nodes. We use two algorithms (i.e., Algorithm 1 for the PrimaryDomain and Algorithm 2 for the SubDomain/PathDomain) to compute the similarity score between an input text string and a string name suggested by the Google suggestion API [49]. Those domain features look similar; however, they are owning different properties, for example a PrimaryDomain cannot be empty as SubDomain/PathDomain, or a phishing PrimaryDomain often contains an IP address.
  Finally, other input values are from domain rank features (such as PageRank, GoogleIndex and AlexaReputation) presented in a previous subsection III-A2.
  Note that some input nodes can be "null" when the system cannot collect sufficient information; for example, values of web traffic features can be empty due to their non-existence on Google or Alexa. Therefore, Fi-NFN allows "null" values in the input vectors. To validate the performance , we also make many case studies (discussed later in Section V) instead of conducting studies of only ideal cases as in existing works [50], [51].
- For the second layer of Fi-NFN, Fig. 8 shows the values of all nodes that are fuzzified underlying the left and right sigmoid membership functions [52]. Depending on each feature of an input node, we use membership functions

Algorithm 1: Calculating the heuristic value of PrimaryDomain

```
Input: d is a PrimaryDomain
Output: The heuristic value of PrimaryDomain
if d is IP then
    //Phishing suggestion
    Return d belongs to a phishing site;
else
    $Result = SuggestionGoogle(d)$;
    if Result is NULL then
        //Legitimate suggestion
        Return d belongs to a legitimate site;
    else
        $value = Levenshtein(Result, d)$;
        Return $value$;
    end
end
```

Algorithm 2: Calculating the heuristic value of SubDomain/PathDomain

```
Input: m is a SubDomain/PathDomain
Output: The heuristic value of
 SubDomain/PathDomain
if m is NULL then
    //Legitimate suggestion
    Return m belongs to a legitimate site ;
else
    $Result = SuggestionGoogle(m)$;
    if Result is NULL then
        //Legitimate suggestion
        Return m belongs to a legitimate site;
    else
        $value = Levenshtein(Result, m)$;
        Return $value$;
    end
end
```

with different settings as follows:

$$L_i(x_i) = \frac{1}{1 + e^{-(x_i - b_i)}}, \tag{5}$$

$$P_i(x_i) = \frac{e^{-(x_i - b_i)}}{1 + e^{-(x_i - b_i)}}, \tag{6}$$

where x_i is the input variable of input node $i, i = 1, 2, ..., 6$ and b_i is a parameter assigned differently for PrimaryDomain, SubDomain, PathDomain, PageRank, GoogleIndex and AlexaReputation.

- The defuzzification is processed from the second layer to the input of the fourth layer. The connections from all legitimate nodes (L_i) in the second layer are linked to node T_1. Similarly, all phishing nodes (P_i) are linked to node T_2. Each node computes the firing strength of the associated rule. This layer gathers the respective values of the phishing and legitimate features. The output values of the nodes are crisp values that indicate the phishing/legitimate percentage of an URL. The calculation is presented as follows:

$$\alpha_1 = \prod_{i=1}^{6} L_i(x_i),$$
$$\alpha_2 = \prod_{i=1}^{6} P_i(x_i). \tag{7}$$

- The fourth layer of Fig. 8 contains two nodes N_1 (Normalization Legitimate) and N_2 (Normalization Phishing), which indicate the normalization of the firing levels. The normalization function is necessary to enhance the training phase [53]. The outputs of these nodes are calculated as follows:

$$\beta_i = \frac{\alpha_i}{\sum_{j=1}^{2} \alpha_j}, i = 1, 2. \tag{8}$$

- The final layer in our model Fi-NFN is the output layer that is calculated by

$$O_I = \sum_{i=1}^{2} w_i \beta_i, \tag{9}$$

where w_i is the weight of the node N_i in the fourth layer. From (9), we apply the following sigmoid function as the activation function for the output node:

$$O_o = f(O_I) = \frac{1}{1 + e^{-O_I}}, \tag{10}$$

where O_o is the output value of Fi-NFN.
As shown in Fig. 9, the shape of the sigmoid function forms the separation threshold between two sets.

Our model is designed to flexibly control the threshold in the fuzzy network layer instead of modifying dozens of rules as in [52]. In Fi-NFN, we only need to adjust b_i and the learning rate to improve the accuracy of phishing identification. Furthermore, we can easily add or remove features in the first layer to adapt to the diversity of phishing websites. In that case, the architecture of Fi-NFN needs to change the vector of input nodes without updating the IF-THEN rule set. The remaining layers and calculations of Fi-NFN are still consistent. For scalability, we define a maximum input vector with n elements, where unassigned inputs are set to 0. Connections between layers 1 and 2 are initialized as a matrix ($n \times 2$), while links of unassigned input features are 0. In the worst case with maximum n features, Fi-NFN can identify a phishing website after $2 \times n + 7$ calculation steps (i.e., $2 \times n$ calculation steps to fuzzify n input features at the second layer, 2 aggregation steps of the third layer, 2 normalization steps at the fourth layer, 2 multiplication steps with the weight parameters, and the activation step at the output node).

B. The training phase of Fi-NFN

In this model, we shall describe the delta learning rule with the sigmoid activation function (9). Suppose given a training dataset:

The system first uses the input vector, $\boldsymbol{x}^{(k)}$, to produce its output vector, $O_o^{(k)}$, based on (5)-(10) and then compares this with the desired output, $y^{(k)}$. For each pair k of input/output,

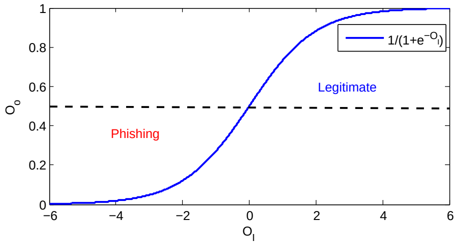


Fig. 9: The sigmoid function shows the threshold to indicate the phishing set and the legitimate set.

TABLE I: The training data set.

No	Input values	Desired output values
1	$\boldsymbol{x}^{(1)} = \{x_1, x_2, ..., x_6\}$	$y^{(1)}$
2	$\boldsymbol{x}^{(2)} = \{x_1, x_2, ..., x_6\}$	$y^{(2)}$
.		
.		
.		
K	$\boldsymbol{x}^{(K)} = \{x_1, x_2, ..., x_6\}$	$y^{(K)}$

we measure the error between the desired output value $y^{(k)}$ and the output $O_o^{(k)}$ as follows:

$$E^{(k)} = \frac{1}{2}(y^{(k)} - O_o^{(k)})^2. \tag{11}$$

Consequently, the summation of the errors in the training dataset is given as follow

$$E = \sum_{k=1}^{K} E^{(k)}. \tag{12}$$

We apply the gradient descent method for updating the weights following the presentation of the input/output pair k (i.e., we minimize the quadratic error function). The update step for the next iteration $k+1$ is represented as follows:

$$\boldsymbol{w}^{(k+1)} = \boldsymbol{w}^{(k)} - R(E^{(k)})', \tag{13}$$

where R is a positive constant called the learning rate. Consider $f(.)$ to be the sigmoid function, then the gradient vector of the error function is calculated as follows:

$$(E^{(k)})' = \frac{d}{d\boldsymbol{w}}\left[\frac{1}{2}(y^{(k)} - \frac{1}{1+e^{-\boldsymbol{w}^T\boldsymbol{\beta}}})\right] \tag{14}$$

$$= (y^{(k)} - O_o^{(k)})O_o^{(k)}(1 - O_o^{(k)})\boldsymbol{\beta}^{(k)}. \tag{15}$$

Therefore, the weight update can be rewritten as follows:

$$\boldsymbol{w}^{(k+1)} = \boldsymbol{w}^{(k)} - R(y^{(k)} - O_o^{(k)})O_o^{(k)}(1 - O_o^{(k)})\boldsymbol{\beta}^{(k)}. \tag{16}$$

We now summarize the training phase of the Fi-NFN model as follows:

- **Step 1**: $R > 0$, $E_{\text{threshold}} > 0$ are chosen.
- **Step 2**: Initialize the weights $\boldsymbol{w}$ at small random values, $k \leftarrow 1$ and the running error $E \leftarrow 0$.
- **Step 3**: The input $\boldsymbol{x}^{(k)}$ and the output $O_o^{(k)}$ are calculated by (5)-(10).
- **Step 4**: The weight $\boldsymbol{w}$ is updated by (16).
- **Step 5**: The cumulative cycle error is calculated by

$$E := E + \frac{1}{2}(y^{(k)} - O_o^{(k)})^2. \tag{17}$$

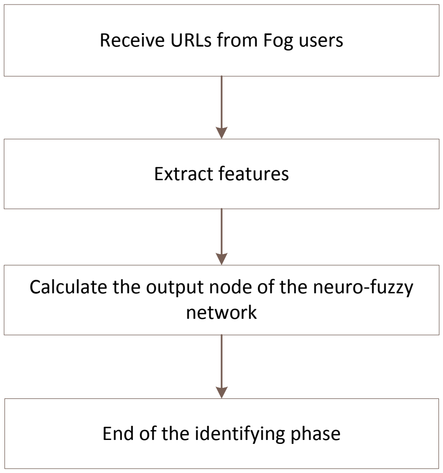

Receive URLs from Fog users
Extract features
Calculate the output node of the neuro-fuzzy network
End of the identifying phase

Fig. 10: The identification phase at a fog node.

- **Step 6**: If $k < K$ then $k \leftarrow k+1$ and then go back to **Step 3**, otherwise go to **Step 7**.
- **Step 7**: Stop the training session if $E < E_{\text{threshold}}$. Otherwise, $E \leftarrow 0$ and go back to **Step 3**.

C. Identification phase for the fog nodes

The training phase is an iterative propagation to update all the weights in Fi-NFN until convergence, meanwhile the identification phase only executes forward the propagation from the input to the output layer. All updated parameters and weights in the training phase are sent and synchronized from the cloud component to all fog nodes, as depicted in Fig. 7. The identification phase is invoked at the fog nodes to detect URL requests from the fog users.

As mentioned in subsection III-B, the phishing identification component operates at a fog node, monitoring and detecting phishing sites for users. Whenever receiving URL requests from fog users, this component extracts features, and then inputs such values into the first layer of Fi-NFN. If the output value of the fifth layer is greater than the identification threshold, such an URL is a legitimate URL, otherwise it is a phishing URL. The flow of the identification phase is presented in Fig. 10.

V. Simulation and Numerical Results

In this section, we discuss our settings and datasets for the evaluation of our model. We implemented and performed side-by-side comparisons with prior works. Furthermore, we have built a test-bed as a fog network to validate the performance of our proposed algorithm.

A. Settings

We collected 11,660 URLs for phishing sites from PhishTank [7] and 10,000 URLs for legitimate sites from DMOZ [54] to make the training and testing datasets in both traditional and fog networks. We then randomly divide this dataset into the training dataset and the testing dataset following the ratio 75:25.

In order to set values for the parameters b_i, we derive the proper settings of parameters (shown in Table III) based on

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2831197, IEEE Transactions on Network and Service Management

TABLE II: The distribution of dataset.

Feature vector size for setting the URL features threshold	$b_i < 3$	$b_i < 4$	$b_i < 5$
11600 (phishing URLs) 10000 (legitimate URLs)	72.143% 83.533%	78.106% 72.006%	82.062% 63.062%
Feature vector size for setting the PageRank threshold	$b_i < 6$	$b_i < 7$	
11600 (phishing URLs)	100%	88.396%	
Feature vector size for setting the AlexaReputation threshold	$b_i < 20$	$b_i < 21$	
11600 (phishing URLs)	100%	82.132%	
Feature vector size for setting the GoogleIndex thresholds	$b_i < 15$	$b_i < 16$	
11600 (phishing URLs)	100%	88.382%	

TABLE III: Parameters b_i for the membership function.

Value	**Description**
4	Used for the PrimaryDomain, SubDomain, PathDomain value.
6	Used for the PageRank value.
20	Used for the AlexaReputation value.
15	Used for the GoogleIndex value.

the observation in the whole dataset. We also present the observation results in Table II) to illustrate how we set the values for b_i.

Other parameters are shown in Table IV. We divide the testing dataset into three URL sets, including the long URL set (i.e., URLs with full features), the short URL set (i.e., URLs that lack features), and the random URL set. Finally, the weights (w_1, w_2) are initialized randomly from -0.5 to 0.5.

B. Results

We conduct the simulation using the above datasets and settings. The convergence, accuracy of phishing identification, and response time are recorded as outputs of our simulation. We compare the performance of Fi-NFN to the current benchmark approaches, such as Fuzzy [24], Neural network [17], Google PageRank [42], eMCAC [22] and FACA [21]. First, we provide a brief outline of those methods that we compare Fi-NFN with:

- Fuzzy: We compare Fi-NFN with an online algorithm that classifies URLs using the fuzzy approach. Essentially, Fuzzy is built by a rule set based on the URL characteristics [24].
- Neural network: Neural network is an approach proposed by [17] using the neural network model to identify phishing URLs. We implement a three-layer model and use our dataset to train this neural network.
- Google: This is the popular tool of Google [20], which can be easily installed on web browsers. GoogleToolbar can detect phishing terms based on input keywords. In this work, to evaluate the performance of our approach, we develop an application that calls the Google API to detect phishing URLs at fog nodes instead of installing them on user devices.
- eMCAC [22] and FACA [21]: These are new approaches based on the rule set method to detect phishing URLs. Similar to the Google API, we implement eMCAC and FACA on fog nodes for detection.

TABLE IV: Parameters of the Fi-NFN model.

Name	**Value**
$E_{\text{threshold}}$	$0.5 \cdot 10^{-5}$
Epoch threshold	1000
Weights (w_1, w_2)	Initialize randomly from -0.5 to 0.5
Identifying threshold	0.5

TABLE V: Training duration.

Size of the training dataset	**Time**
5000 URLs	985 s
6000 URLs	1141 s
7000 URLs	1342 s
8750 URLs	1498 s

1) Evaluation in a traditional environment:

Convergence. To evaluate the convergence of Fi-NFN, we run with different learning rates from 0.1 to 0.9. Fig. 11 shows the convergence of our model after more than 800 epochs. Corresponding to our setting, we observe that our model obtained the most rapid convergence and the lowest error with the learning rate $R = 0.7$. Furthermore, we evaluate the duration of the training phase for different sizes of the dataset. The results in Table V show that the training duration does not increase exponentially when increasing the number of sites.

Accuracy. We first evaluate the impact of various learning rates on Fi-NFN. The accuracy results are given in Fig. 12, where we use three types of URLs. Our model achieves the best average accuracy when the learning rate R is set by 0.7.

In another evaluation, we use three measurements as follows:

- Accuracy: The rate of websites that are correctly detected, calculated by: $\frac{\#\text{True phishing}+\#\text{True legitimate}}{\text{Total sites}}$.
- Sensitivity: The rate of legitimate websites that are correctly detected, calculated by: $\frac{\#\text{True legitimate}}{\#\text{True legitimate}+\#\text{False alarm websites}}$.
- Specificity: The rate of phishing websites that are correctly detected, calculated by: $\frac{\#\text{True phishing}}{\#\text{True phishing}+\#\text{Missed detection websites}}$.

The results in Fig. 13 show the high values of Accuracy, Specificity, and Sensitivity. The figure also illustrates somewhat inconsistent performance for the short URL type, which often has empty input values.

In detail, short URLs contain two types of URLs, one type has full traffic history, such as the values of PageRank,

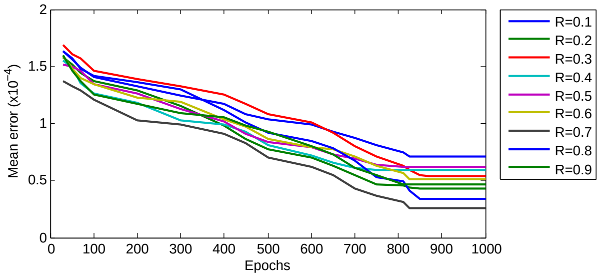


Fig. 11: Convergence of Fi-NFN.

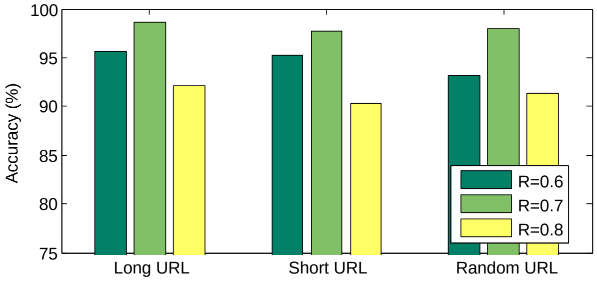


Fig. 12: Evaluation of the accuracy for different learning rates R.

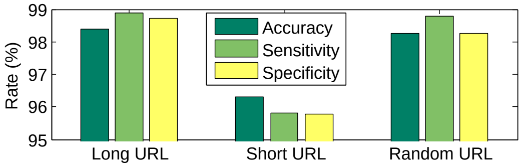


Fig. 13: Performance of Fi-NFN with different measurements.

AlexaReputaion, and Google Index, while another type lacks any of this information. By combining various independent features to classify URLs, we can enhance the identification performance. Fig. 14 shows a significant improvement when using multiple features in phishing identification as compared to only using domain features (denoted as Domain in the legend) or only using web-traffic features (denoted as Web-traffic in the legend). We also compare our method with Fuzzy and Neural network. To evaluate, we create 132 testing sets that are randomly chosen from the dataset above with 1000 URLs in each set. We show results of these comparisons in Fig. 15. In particular, the snapshot of all testing result is shown in Fig. 15a, where our method illustrates a very high performance. Specifically, our method has an average accuracy of 98.36%, while Fuzzy obtains an average accuracy rate of 88.19% and Neural network has an average accuracy rate of 95.61%. Moreover, Fig. 15b illustrates not only the high accuracy, but also the stable characteristics of our method. The median block of our method is the smallest, meaning that the average accuracy among 132 testing sets is more stable than those of Fuzzy and Neural network.

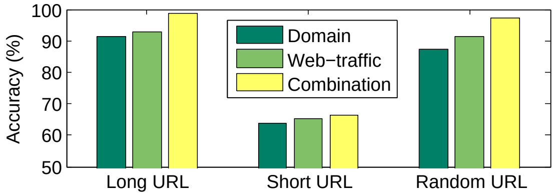


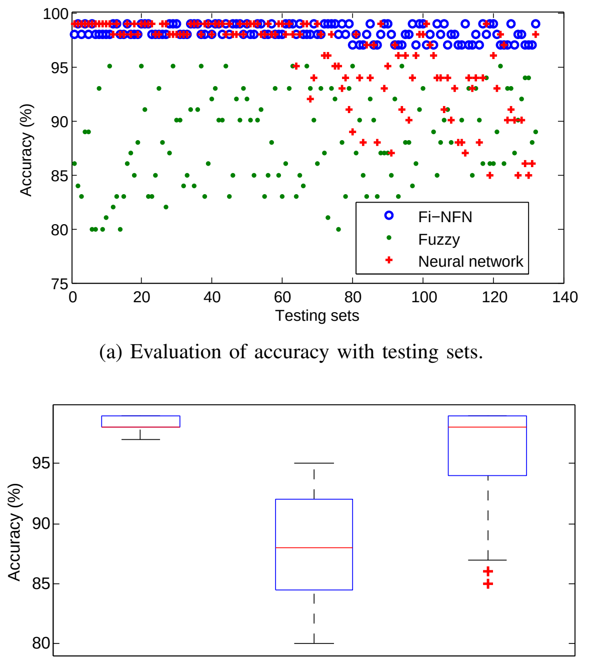
Fig. 14: Evaluation of combination features in phishing identification.

To illustrate three cases of phishing identification, such as long URLs, short URLs, and random URLs, we make a comparison with state-of-the-art methods to evaluate our model, including Neural network, Fuzzy, eMCAC and FACA. We choose these approaches for comparison since i) they can be implemented to protect the client side, ii) they can respond in real time, and iii) they are related to our approach in terms of detection features. By using Fuzzy, the accuracy is not stable since it depends on the definition of the rule set. Even though eMCAC and FACA are new methods using rule sets, they still achieve low performance in our testing datasets. Actually, for eMCAC and FACA, we implement rule sets that are different from their works because our dataset is only used for URL features. We build the rule set as mentioned in Table III and in [22], then use their algorithms to detect phishing URLs. Without using the full features as in [21], [22], these approaches cannot achieve high performance in all cases, as shown in Fig. 16. Furthermore, entirely based on the rule sets, Fuzzy, eMCAC and FACA are all strongly sensitive to the rule set definitions and the thresholds. This performance degradation is illustrated in Fig. 16b when we increase the threshold b_i (it leads to change policies in the rule set). For Fi-NFN, it looks more stable and performs better than Neural network and Fuzzy in terms of accuracy measurement, even when the threshold b_i is increased, as shown in Fig. 16. With short URLs, Fi-NFN outperforms the others since the neuro-fuzzy network can learn similar URLs from the training set. Moreover, Fi-NFN is not strongly sensitive to the setting parameters after training, when we change parameters b_i.

2) Implementation and testing on a fog node: We setup one testbed for the simulation on a fog network with one powerful computing signboard, Odroid-XU3 [55] to act as a fog node. The detailed configuration of the fog node is as follows: Android OS v4.4, Exynos 5422 Octa 1.8GHz, Mali-T628 MP6, 2GB RAM, 32 GB storage, Wi-Fi 802.11 a/b/g/n, and Wi-fi Direct. To measure the performance of our model in fog networks, we implement Fi-NFN as an application on the fog node. We also implement Neural network, Fuzzy, FACA and Google API methods on the fog node to compare with Fi-NFN. Especially for Google API, we develop an application that calls Google API to detect phishing URLs instead of installing GoogleToolbar application on user devices. We then run the simulations during 10 hours with automated agents, as demonstrated in Fig. 17. These agents automatically and continuously send URL requests to the fog node. When receiving URL requests, the fog node automatically invokes the identification phase to detect URLs. The URL requests of each agent are extracted randomly from the testing dataset as mentioned before.

We make a network traffic measurement to compare Fi-NFN with Google. Our approach consumes less traffic per each detection compared to Google, even for updating. The result of our simulation in Table VI illustrates traffic efficiency of Fi-NFN.

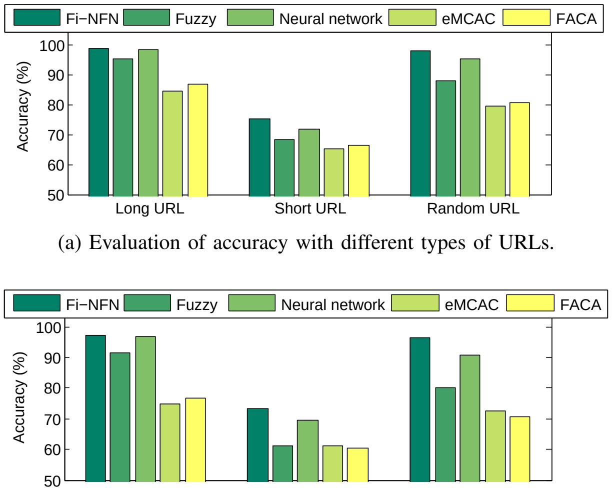
We measure three features to evaluate our model, including response time, error rate and accuracy. First, we evaluate the



(a) Evaluation of accuracy with testing sets.

(b) Evaluation of the median accuracy.

Fig. 15: Comparison of our method Fi-NFN with Fuzzy and Neural network.



(a) Evaluation of accuracy with different types of URLs.

(b) Evaluation of accuracy with a different setting of b_i.

Fig. 16: Comparison of our method Fi-NFN with state-of-the-art methods.

TABLE VI: Network traffic measurement.

Name	Value
Fi-NFN	3.528 KB per training phase 0.875 KB per request.
Google Toolbar	6.383 KB per request.

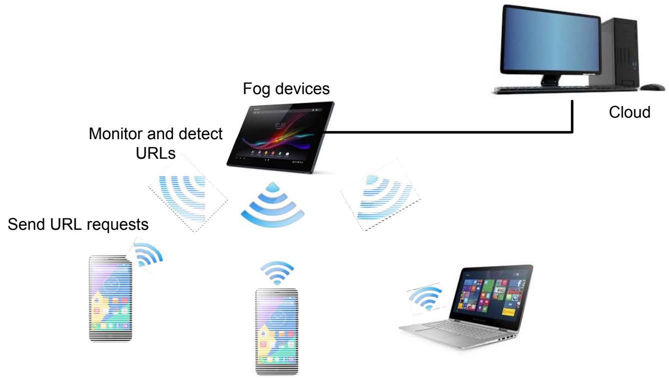


Fig. 17: The fog network testbed.

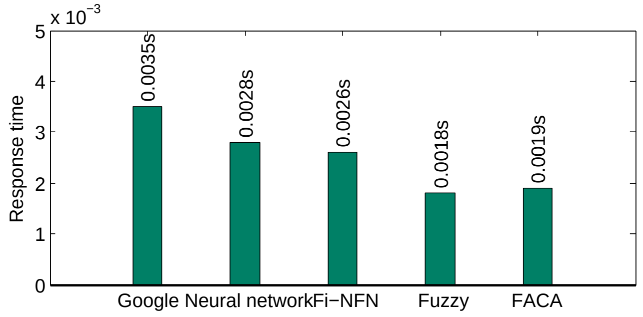


Fig. 18: Evaluation of the response time (duration of time from sending an URL request until receiving a response at the user device).

response time by comparing Fi-NFN to other approaches. The results of Fi-NFN are similar to those of Neural network, as shown in Fig. 18. For Fuzzy and FACA, considering detection based on the rule sets, the response times are faster than that of Fi-NFN and Neural network. In our test case, we measure the response time under a good network condition (no network congestion).

Second, we evaluate the accuracy of phishing detection by measuring the error rate, the missed detection, and the false alarm of all agents in the fog network for 10 time slots. Fig. 19 presents the detection result of Fi-NFN, which is better than other approaches. The average error rate of Fi-NFN from the first to the tenth timeslot is in a range from 1.32658% to 1.98724%, which demonstrates the stable performance of our system compared to other approaches. For missed detections and false alarms, Fig. 20 shows the average rate of each approach. Fi-NFN achieves the low rate of missed detections as well as false alarms during the testing period. Compared to others, the false alarm rate in Google is very low, but its missed detection rate is very high. For Neural network, accurate detection can be achieved with trained URLs excluding new URLs or URLs with empty features. The results of Fuzzy and FACA are similar, showing they do not work well with the short URL set. Considering both missed detection and false alarm rates, Fi-NFN outperforms the other methods.

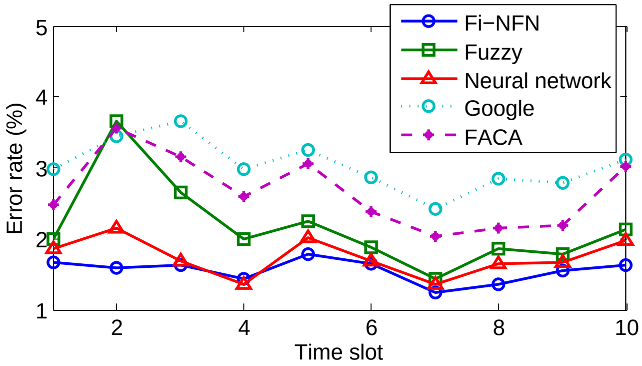


Fig. 19: Evaluation of the error rate for 10 time slots with the fog test-bed.

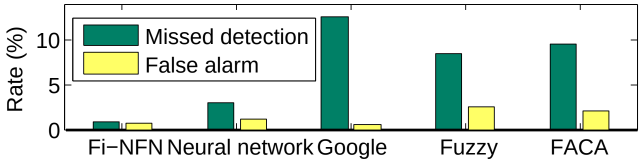


Fig. 20: Comparison of the missed detection and false alarm rates between FACA, Neural network, Google and Fi-NFN.

VI. Conclusion

In this paper, we consider the security issues regarding the fog network to enhance network safety. In particular, we study the phishing website problem and propose an identification architecture on the fog network. Based on the advantages of the fog architecture and the neuro-fuzzy approach, we propose a phishing identification model, called Fi-NFN, to protect local devices easily and quickly. Without consuming many resources from local devices, our Fi-NFN model not only transparently protects users in real time, but also improves the quality of services at the edge of the network. Without using an inefficient blacklist method, we design a five-layer neuro-fuzzy network with six heuristic input values (PrimaryDomain, SubDomain, PathDomain, PageRank, GoogleIndex and Alexareputation). Our simulation results indicate that the efficiency of phishing identification after training with the training dataset by improving the average accuracy to 98.36% and reducing the missed detection and false alarm rates to 0.9% and 0.74 %, respectively. We also compare our approach with current methods [17], [20] and [24] to evaluate our model. Simulation results show that our method is more efficient, stable and accurate. Especially, various testing results indicate that our model in a fog computing environment is not only possible, but also can be applied practically.

References

[1] L. Wenyin, G. Huang, L. Xiaoyue, X. Deng, and Z. Min, "Phishing web page detection," in *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*. IEEE, pp. 560–564.

[2] P. Stavroulakis and M. Stamp, *Handbook of Information and Communication Security*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[3] Anti-phishing working group. Accessed Sept 2016. [Online]. Available: http://www.antiphishing.org.

[4] Mobile marketing statistics. Accessed Mar, 2017. [Online]. Available: http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/

[5] Phishing attacks. Accessed Sept 2015. [Online]. Available: https://securityintelligence.com/

[6] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: a content-based approach to detecting phishing web sites," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 639–648.

[7] PhishTank. Accessed Nov 2015. [Online]. Available: http://www.phishtank.com/stats/2014/01/

[8] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, J. Hong, and C. Zhang, "An empirical analysis of phishing blacklists," in *Proceedings of Sixth Conference on Email and Anti-Spam (CEAS)*, 2009.

[9] V. M. Gandhimathi K1, "Identifying similar web pages using scoring methods for web community mining," in *Proc. of Int. Conf. on Advances in Computer Science, AETACS*, 2013.

[10] B. B. Gupta, A. Tewari, A. K. Jain, and D. P. Agrawal, "Fighting against phishing attacks: state of the art and future challenges," in *Neural Computing and Applications*, 2016.

[11] Apwg reports. [Online]. Available: http://docs.apwg.org/reports/

[12] Fog computing and the internet of things: Extend the cloud to where the things are. [Online]. Available: http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf

[13] Y. Li and M. Chen, "Software-Defined Network Function Virtualization: A Survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[14] Real time anti-phishing. Accessed on Sept 2016. [Online]. Available: http://www.brightcloud.com/services/real-time-anti-phishing.php

[15] Anti Spam Hardware. Accessed Sept 2016. [Online]. Available: http://www.windowsnetworking.com/hardware/Anti-Spam-Hardware/modusGate-Anti-Spam-Appliance.html

[16] Security Intelligence Blacklisting. Accessed Sept 2016. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/security/firepower/601/configuration/guide/fpmc-config-guide-v601/Security_Intelligence_Blacklisting.pdf

[17] N. Zhang and Y. Yuan, "Phishing detection using neural network, cs229 lecture notes." [Online]. Available: http://cs229.stanford.edu/proj2012/ZhangYuan-PhishingDetectionUsingNeuralNetwork.pdf,2012

[18] G. Xiang, J. Hong, C. P. Rose, and L. Cranor, "Cantina+: a feature-rich machine learning framework for detecting phishing web sites," *ACM Transactions on Information and System Security (TISSEC)*, vol. 14, no. 2, pp. 1–28, 2011.

[19] Vulnerability in spotify android-app. Accessed on Sept 2015. [Online]. Available: http://blog.trendmicro.com/trendlabs-security-intelligence/vulnerability-in-spotify-android-app-may-lead-to-phishing/

[20] Google safe browsing. Accessed Sept 2016. [Online]. Available: https://safebrowsing.google.com/

[21] N. Abdelhamid, "Multi-label rules for phishing classification," *Applied Computing and Informatics*, vol. 11, no. 1, pp. 29 – 46, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210832714000210

[22] W. Hadi, F. Aburub, and S. Alhawari, "A new fast associative classification algorithm for detecting phishing websites," *Applied Soft Computing*, vol. 48, pp. 729 – 734, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494616303970

[23] L. . T. C. James, J. ; Sandhya, "Detection of phishing urls using machine learning techniques," in *Control Communication and Computing (ICCC), 2013 International Conference on*, 2013, pp. 304–309.

[24] M. Aburrous, M. A. Hossain, K. Dahal, and F. Thabtah, "Intelligent phishing detection system for e-banking using fuzzy data mining," *Expert systems with applications*, vol. 37, no. 12, pp. 7913–7921, 2010.

[25] A. N. V. Sunil and A. Sardana, "A pagerank based detection technique for phishing web sites," in *Computers & Informatics (ISCI), 2012 IEEE Symposium on*. IEEE, 2012, pp. 58–63.

[26] R. Verma and K. Dyer, "On the character of phishing urls: Accurate and robust statistical learning classifiers," in *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '15. New York, NY, USA: ACM, 2015, pp. 111–122.

[27] L. Wenyin, G. Liu, B. Qiu, and X. Quan, "Antiphishing through phishing target discovery," *IEEE Internet Computing*, vol. 16, no. 2, pp. 52–61, March 2012.

[28] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, *Fog Computing: A Platform for Internet of Things and Analytics*. Springer International Publishing, 2014, pp. 169–186.

[29] IoT, from cloud to fog computing. Accessed Sept 2015. [Online]. Available: http://blogs.cisco.com/perspectives/iot-from-cloud-to-fog-computing

[30] J. Shropshire, "Extending the cloud with fog: Security challenges & opportunities," in *Information Systems Security, Assurance, and Privacy Track Conference*, 2014.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2831197, IEEE Transactions on Network and Service Management

[31] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues," in *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*. IEEE, pp. 1–8.

[32] Network appliance issues advisory for customers facing phishing attacks. Accessed Sept 2016. [Online]. Available: http://www.netapp.com/us/company/news/press-releases/

[33] Cloud NFV White Paper. Accessed date Sept 2016. [Online]. Available: http://cloudnfv.com/WhitePaper.pdf

[34] WHOIS. Accessed Sept 2016. [Online]. Available: http://www.whois.com/

[35] S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in *Proceedings of the 2007 ACM workshop on Recurring malcode*. ACM, 2007, pp. 1–8.

[36] Spam and open-relay blocking system. Accessed Sept 2016. [Online]. Available: http://www.sorbs.net

[37] Url blacklist. Accessed Sept 2016. [Online]. Available: http://uribl.com

[38] Surbl. Accessed Sept 2016. [Online]. Available: http://www.surbl.org

[39] Symantec norton. Accessed Sept 2016. [Online]. Available: http://www.symantec-norton.com/

[40] R. Dhamija and J. D. Tygar, "The battle against phishing: Dynamic security skins," in *Proceedings of the 2005 symposium on Usable privacy and security*. ACM, 2005, pp. 77–88.

[41] J. H. Huh and H. Kim, "Phishing detection with popular search engines: Simple and effective," in *Foundations and Practice of Security*. Springer, 2012, pp. 194–207.

[42] Checking Page Rank. [Online]. Available: https://www.prchecker.info/check_page_rank.php

[43] Link analysis. Accessed Sept 2015. [Online]. Available: http://snap.stanford.edu/class/cs224w-readings/borodin05pagerank.pdf

[44] Alexa. [Online]. Available: http://tutology.net/category/how-php/get-alexa-rank-php-and-alexa-api

[45] Google index. Accessed Sept 2016. [Online]. Available: "https://www.google.com/insidesearch/howsearchworks/crawling-indexing.html"

[46] J.-S. R. Jang, "Anfis: adaptive-network-based fuzzy inference system," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 23, no. 3, pp. 665–685, 1993.

[47] R. Fullér, "Neural fuzzy systems," 1995.

[48] P. Liu and H. Li, "Fuzzy neural networks for storing and classifying," in *Fuzzy Neural Network Theory And Application*. World Scientific, 2004, pp. 25–67.

[49] Query suggestion service of google. Accessed Sept 2016. [Online]. Available: "http://www.google.com/support/enterprise/static/gsa/docs/admin/70/gsa_doc_set/xml_reference/query_suggestion.html"

[50] H. Zhang, G. Liu, T. W. Chow, and W. Liu, "Textual and visual content-based anti-phishing: a bayesian approach," *Neural Networks, IEEE Transactions on*, vol. 22, no. 10, pp. 1532–1546, 2011.

[51] W. Zhuang, Q. Jiang, and T. Xiong, "An intelligent anti-phishing strategy model for phishing website detection," in *Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on*. IEEE, 2012, pp. 51–56.

[52] J.-S. Jang and C.-T. Sun, "Neuro-fuzzy modeling and control," *Proceedings of the IEEE*, vol. 83, no. 3, pp. 378–406, 1995.

[53] R. Rojas, *Neural networks: a systematic introduction*. Springer Science & Business Media, 1996.

[54] Dmoz. Accessed Nov 2015. [Online]. Available: http://rdf.dmoz.org/rdf/

[55] Hard kernel. Accessed Sept 2015. [Online]. Available: http://www.hardkernel.com/main/products/prdtinfo.php?gcode=G140448267127

Chuan Pham He received the BS degree from Hochiminh City University of Transport, master degree from HoChiMinh City University of Sciences and Ph.D degree from Kyung Hee University, in electrical and computer engineering, in 2004, 2008 and 2017, respectively. He was a postdoc fellow in Department of Computer Science and Engineering, Kyung Hee University from August, 2017. Since 2018, he has been a postdoc fellow in Synchromedia - École de Technologie Suprieure, Université du Québec. His research interest is applying analytic techniques of optimization and machine learning to network applications in terms of cloud and mobile-edge computing, datacenters, resource allocation for virtual networks, and Internet of Things.

Luong A. T. Nguyen He is the head of Information System Department, Ho Chi Minh City University of Transport, Vietnam. He received B.Sc. and M.Sc. in Computer Science from University of Science Viet Nam National University Ho Chi Minh City. He also received Ph.D. degree in Control Engineering and Automation. His current research interests include intelligent control, fuzzy systems, neural network, security and cloud computing.

Nguyen H. Tran He received the BS degree from Hochiminh City University of Technology and Ph.D degree from Kyung Hee University, in electrical and computer engineering, in 2005 and 2011, respectively. He was an Assistant Professor with Department of Computer Science and Engineering, Kyung Hee University from 2012 to 2017. Since 2018, he has been with the School of Information Technologies, The University of Sydney, where he is currently a Senior Lecturer. His research interest is applying analytic techniques of optimization, game theory, and machine learning to cutting-edge applications such as cloud and mobile-edge computing, datacenters, resource allocation for 5G networks, and Internet of Things. He received the best KHU thesis award in engineering in 2011 and several best paper awards, including IEEE ICC 2016, APNOMS 2016, and IEEE ICCS 2016. He has been the Editor of IEEE Transactions on Green Communications and Networking since 2016.

Eui-Nam Huh He is a professor in Department of Computer Science and Engineering. His interesting research areas are: Cloud Computing, Ubiquitous Computing, High Performance Network, Sensor Network, Distributed Real Time System, Grid, Network Security. He received Masters degree in Computer Science from University of Texas, USA in 1995 and Ph. D degree from the Ohio University, USA in 2002. He has also served for the WP-DRTS/IPDPS community as program chair in 2003. He has been an editor of Journal of Korean Society for Internet Information and Korea Grid Standard group chair since 2002. He was also an Assistant Professor in Seoul Womens University, South Korea

Choong Seon Hong received the B.S. and M.S. degrees in electronic engineering from Kyung Hee University, Seoul, South Korea, in 1983 and 1985, respectively. He received the Ph.D. degree in computer engineering from Keio University, Tokyo, Japan, in March 1997. In 1988, he joined KT, where he worked on broadband networks as a Member of the Technical Staff. In September 1993, he joined Keio University, Japan. Since September 1999, he has been a Professor in the Department of Computer Science and Engineering, Kyung Hee University, Seoul, Korea. His research interests include future Internet, ambient intelligent technology, wireless networks, network security, and network management.

Dr. Hong served as a Program Committee Member and an Organizing Committee Member for international conferences such as NOMS, IM, APNOMS, and ICOIN. He is a Member of the ACM, IEICE, IPSJ, KIISE, KICS, and KIPS.