Bank Web Sites Phishing Detection and Notification System Based on Semantic Web technologies

Faisal Alkhateeb, Ahmed M. Manasrah and Abed Al Raoof Bsoul
Faculty of Information Technology and Computer Science
Yarmouk University
Irbid 21163, Jordan.
{alkhateebf, ahmad.a, raoofbsoul}@yu.edu.jo

Abstract

Phishing is an online theft of sensitive information that swindles innocent users into disclosing private information such as user names, passwords, and credit card numbers. The reported number of phishing attacks is growing daily, hence, the loss of the resulting damages are escalating. As a result, there is an urgent need for anti-phishing solutions that is arisen by researchers as well as the IT industry worldwide. Although a number of solutions to mitigate phishing attacks have been proposed, yet they still suffer from high false positive and negative results as well as questioning the feasibility of their implementation.

In this work, we propose a system for client-side defenses such as browser plug-ins and classification techniques that are adopted in such detection scenarios. The system inspects the HTML pages as an annotated document represented or embedded in XHTML format using RDF annotations. While the proposed solution has been tested using real sites acquired from the World Wide Web and government agencies concerned on the problem , the method has a better detection accuracy that reaches 96% while false positive rate decreased to 4%. The results show a promising findings in the area of phishing detection that requires hand-in-hand collaboration between various banking branches and the country's central or authorized bank. Additionally, the system notifies the corresponding bank about the phishing web sites, and the bank in turn notifies its clients.

Keywords**:** *Phishing, RDFa, RDF, SPARQL.*

1. Introduction

The cyberspace is a constant arena for both red and black hat users. One of the cyberspace challenging and continuous tribulations is phishing attacks. In which innocent users are deceived to disclose their personal and confidential information such as user names, passwords, and credit card information by directing them to enter these details into fake websites whose look and feel is similar to the legitimate one. On the other hand, phishers are always moving toward developing targets resting sure that their users or administrators are not very aware of the

available security vulnerabilities, unlike the well-defended Internet assets that are not following the default installation and configuration. This was observed by the APWG group in their report in late 2010, with implications for phishing targets, service providers, and anti-phishing responders [1]. The same report aims to quantify the scope of the global phishing problem through examining all the reported phishing attacks in the second half of 2010 (2H2010, or July 1, 2010 through December 31, 2010) [1]. The data collected by the Anti-Phishing Working Group and supplemented by several phishing feeds and private sources is summarized in Table 1.

Table 1. Basic Statistics

	2H2010	1H2010	2H2009	1H2009	2H2008
Phishing domain names	42,624	28,646	28,775	30,131	30,454
Attacks	67,677	48,244	126,697	55,698	56,959
IP-based phish (unique IPs)	183	177	173	171	170
Maliciously registered domains	2,318	2,018	2,031	3,563	2,809
IDN domains	11,769	4,755	6,372	4,382	5,591
	10	10	12	13	10

In spite of the many anti-phishing solutions, the phishing phenomena has continued to thrive. Hence, confidence on automatic tools to detect and prevent phishing attacks has decreased; because relying on such automatic ways to pin point the phishing attacks makes these tools suffer from both false positive and false negative. For example, attackers can acclimatize themselves to automatic spam and phishing email filters and find new ways to evade them.

In this paper, we present a novel method that relies on the website structures and features (i.e. bank name, branch name, base URL, address) represented in RDF format to decide on its legitimacy. The RDFa allows XHTML authors to mark up human-readable data with machine-readable indicators for browsers and other programs to interpret. The proposed solution urged that each bank profile should be activated and added to the RDF knowledge base once the bank took the opening authorization from the country Central Bank. An RDF extractor is used to extract RDF features from the requested web page of the bank before the user view the website on his browser to avoid any scripting hooks. The extracted RDF features will be processed as RDF ontology and fed into the system for decision making.

Paper Outline. The remainder of the paper is organized as follows. In Section 2, we introduce RDF and the SPARQL language. Section 3 is dedicated to the proposed system. We present the experimental results of the proposed system in Section 5. We discuss other related work in Section 6. Finally, we conclude our work in Section 7.

2. Preliminaries

This section provides an overview of the elements that are necessary for presenting the proposed approach namely: RDF and SPARQL. RDF is basically a graph-based language, which makes it very suitable for representing social networks (as they have a graph nature) while SPARQL is used for searching RDF data.

2.1 RDF

RDF (Resource Description Framework) [2] is a knowledge representation language dedicated to the annotation of resources within the Semantic Web. Currently, many documents are annotated via RDF due to its simple data model and its formal semantics. In its abstract syntax, an RDF document is a set of triples of the form `<subject, predicate, object>`.

The following example means that there exists a resource named "Arab Bank" whose homepage, phone and fax are <http://www.arabbank.com.jo/>, "+962-6-5694901" and "962-6-56949141", respectively. Note that `foaf`[1] and `vcard`[2] (which are used in this paper), are two RDF vocabularies.

Example 1 *The assertion of the following RDF triples:*

```
{ _:b1 foaf:name "Arab Bank" .
  _:b1 foaf:homepage <http://www.arabbank.com.jo/> .
  _:b1 foaf:phone" "+962-6-5694901" .
  _:b1 vcard:fax" "+962-6-56949141" .
}
```

An RDF document can be represented by a directed labeled graph, as shown in Figure 1, where nodes are terms appear as the subject or object in a triple and arcs are the set of triples (*i.e.*, if `<s, p, o>` is a triple, then $s \xrightarrow{p} o$).

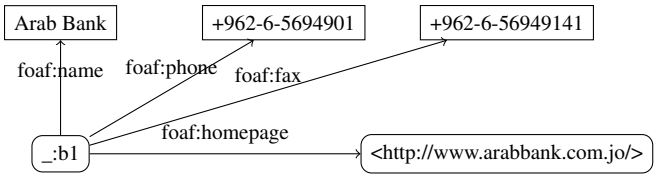

Figure 1. An RDF graph.

RDF annotations can be represented in several formats and it can be embedded in HTML or XHTML documents using RDFa annotations [3].

[1] `xmlns.com/foaf/0.1/`
[2] `www.w3.org/TR/vcard-rdf/`
[3] `http://www.w3.org/TR/xhtml-rdfa-primer/`

2.2 SPARQL

SPARQL is a W3C recommendation language developed in order to query RDF knowledge bases, *e.g.* to retrieve nodes from RDF graphs [3]. A simple SPARQL query is expressed using a form resembles standered query language (SQL) SELECT statement:

$$\texttt{SELECT}\ \vec{B}\ \texttt{FROM}\ u\ \texttt{WHERE}\ P$$

where u is the URL of an RDF graph G to be queried, P is a SPARQL graph pattern (*i.e.*, a pattern constructed over RDF graphs with variables) and $\vec{B}$ is a tuple of variables appear in P. Intuitively, an answer to a SPARQL query is an instantiation of the variables of $\vec{B}$ by the terms of the RDF graph G such that the substitution of the values to the variables of P yields to a subset of the graph G[4].

Example 2 *The following SPARQL query:*

```
SELECT  ?name ?phone
FROM  <Figure1>
WHERE
    ?p1 foaf:homepage <http://www.arabbank.com.jo/> .
    ?p1 foaf:name ?name .
    ?p1 foaf:name ?phone .
```

can be used to retrieve the name and the phone of a bank whose homepage is <http://www.arabbank.com.jo/>. When evaluated against the RDF graph of Figure 1, the following answers will be returned:

#	$?name$	$?phone$
1	$ArabBank$	$" + 962 - 6 - 5694901"$

Another form of SPARQL queries is the ASK query. ASK SPARQL query does the same as the SELECT query, but instead of retrieving the matching for variables, they returns true if there is at least one answer for P in the RDF graph u; otherwise false is returned.

$$\texttt{ASK FROM}\ u\ \texttt{WHERE}\ P$$

3. RDF-Based Phishing Detection System

3.1 System components

Basically, the system is composed of the following components:

- RDF knowledge base:

  This RDF knowledge base contains a profile for each bank branch. Each branch bank profile is formed as a set of RDF triples to represent information about the bank such

[4]When using RDFS semantics [4], this intuitive definition is irrelevant and one can apply RDFS reasoning rules to calculate answers over RDFS documents.

as bank name, branch name, base URL, address, allowed ports or allowed IP addresses. The bank profile is activated and added to the RDF knowledge base once the bank takes the opening authorization from Jordan central bank. This RDF knowledge base is maintained and stored on a server in Jordan central bank and can be updated by authorized persons only. Note that each country can store an RDF knowledge base containing legitimate banking web sites. In this case, an approach for federated SPARQL queries can be adopted [5][5].

```
<bank-id> foaf:name "Arab Bank" .
...
```

Figure 2. An RDF graph representing part of a bank profile.

- An RDFa extractor:

  The RDFa extractor is used to extract RDF features from the bank's web page to be tested. There are several good RDFa extractors. In the proposed system, the online RDFa extractor[6] is used.

- A hash table:

  The hash table contains the names of Jordan banks and their branches indexed by their base URLs.

- SPARQL engine:

  We have used Jena[7] framework which is an inference engine for answering SPARQL queries over RDF knowledge base.

3.2 System work flow

The system works as follow:

1. The URL of the web page of the bank to be tested is provided to the system.
2. The bank features represented in the HTML or XHTML page as RDFa annotations are extracted using the RDFa extractor, and an RDF ontology for the bank profile is constructed and fed to the system.
3. If an IP address is used as a link page then add an RDF triple with IP address to the bank profile and goto step 4. The base URL in the bank profile RDF ontology is compared to the base URL of the requested page. If they are not matched then goto step 5.
4. The bank RDF profile ontology is then converted to a SPARQL query. The SPARQL query is used to search for the bank profile in the RDF ontology of all banks. If a match is found, then the tested web site is legitimate and goto step 6.

[5] www.w3.org/TR/sparql11-federated-query/
[6] http://www.w3.org/2007/08/pyRdfa/
[7] http://jena.sourceforge.net/

5. In this case, the tested web site is not legitimate and a warning message is sent to Jordan central bank about the requested bank to take an action. Also, the corresponding bank will be notifies, which in turn notifies its clients about the phishing web site.

6. Stop

3.3 Case study

To illustrate the behavior of the proposed system, we assume that the client received the link for the web site of Figure 3 for a given bank. For simplicity and without loss of generality, we put the necessary parts in the web site for illustrating the web site.

```
Arab Bank
Tel: +962-6-5694901
Fax: +962-6-5694914
...
```

Figure 3. A part of a bank web page.

The web page of the bank should be annotated with RDFa as illustrated in Figure 4.

```
<body>
 <div xmlns:foaf="xmlns.com/foaf/0.1/"
      xmlns:vcard= "7.http://www.w3.org/2006/vcard/ns#">
  <h1><a property="foaf:name" rel="foaf:homepage"
  href="http://www.arabbank.com.jo//">Arab Bank</a></h1>
   <br/>
   <h2>Tel: <a property="foaf:phone">+962-6-5694901</a></h2>
   <br/>
   <h2>Fax: <a property="vcard:fax">+962-6-5694914</a></h2>
 </div>
</body>
```

Figure 4. RDFa annotations in the (X)HTML document for the BibTeX element of Figure 3.

The RDFa annotations will be extracted to give the following RDF ontology.

```
@prefix foaf: <xmlns.com/foaf/0.1/> .
vcard: <http://www.w3.org/2006/vcard/ns#> .
_:b1 foaf:name "Arab Bank" .
_:b1 foaf:homepage <http://www.arabbank.com.jo/> .
_:b1 foaf:phone" "+962-6-5694901" .
_:b1 vcard:fax" "962-6-56949141" .
```

This RDF ontology is then transformed to an ASK SPARQL query (see below), which will be evaluated against the RDF ontology of all banks profiles. If a match exists then this query

returns TRUE (*i.e.*, the link is for legitimate web site), FALSE otherwise (*i.e.*, the link is for a phishing web site). It should be noticed that some of the above bank attributes will be used by the system to notify the corresponding bank.

```
PREFIX foaf: <xmlns.com/foaf/0.1/> .
       vcard: <http://www.w3.org/2006/vcard/ns#> .
ASK
FROM  <bank-profiles>
WHERE
   _:b1 foaf:name "Arab Bank" .
   _:b1 foaf:homepage <http://www.arabbank.com.jo/> .
   _:b1 foaf:phone" "+962-6-5694901" .
   _:b1 vcard:fax" "962-6-56949141" .
```

4. Types of Phishing Attacks

Since the cyberspace is open for all limitlessly, it opens the doors for users who are seeking certain gains (i.e. financials) to create methods and ways to deceive users to reveals certain confidential information such as credit card numbers, password...etc. Some of the used methods is phishing which aims to direct users accessing online websites into another website which is identical to the original and authenticated website, but under the control of the user / hacker. Phishers (users originate / controlling phishing website) are always evolving in their phishing attacks in an attempt to evade the known phishing detection methods, some of their known phishing types that have been identified and widespread are as follow:

1. *Spoofing emails and websites.*

   Phishing attacks initially starts with sending out phishing emails (i.e. spoofed emails) to a group of users as an attempt to deceive them to reveal their passwords and account information. Despite the fact that this type of phishing attack is known to the majority of the internet users, still this type of attack might find it's away to the users eventhough they have learned not to share sensitive information over e-mails. Therefore, phishers (attackers) start to employ more advanced and sophisticated techniques consisting of a blend of spoofed e-mails and web sites to deceive their victims. This new era of phishing attacks depends on sending out phishing or spoofed emails to a large number of users that appears as if it is from a genuine organization such as a bank or trusted users asking for profile and information updates or email verifications. Upon clicking on the embedded link within the spoofed email, the user (victim) is redirected to a website under the command of the attacker that looks exactly the same as the original website to increase the victim confidence to enter his username/password that will be send directly to the attacker.

2. *Malware based phishing attacks.*

   refers to scams that involve running malicious software on users' PCs. Malware can be introduced as an email attachment, as a downloadable file from a web site, or by exploiting known security vulnerabilities. Typically, the malware (i.e. virus, worm, bot) runs quietly in the background, collecting personal data that users enter on unsecured web sites. However, Malware based phishing attacks are not our focus, but to mitigate web

site-based phishing attacks that aim to deceive victims into giving away their credential information.

3. *Content injection phishing*

   describes the situation where hackers replace part of the content of a legitimate site with false content designed to mislead or misdirect the user into giving up their confidential information to the hacker. For example, hackers may insert malicious code to log user's credentials or an overlay which can secretly collect information and deliver it to the hacker's phishing server, such as the use of JavaScript hook that read whatever the user keys into the form fields and transfer these info back to the hacker.

5. Testing and Evaluation Results

Our dataset consists of a mixof phishing and legitimate URLs from the period Jan 2010 to Jan 2012. Our sources for phishing URLs are PhishTank[8]. Our sources for legitimate URLs are Yarmouk University's HTTP access logs. Our dataset is made of 13,208 unique legitimate URLs and 13,208 unique phishing URLs (26,416 URLs in total). This make is a 1:1 Phish dataset in order to make our analysis unbiased. We extracted 13,208 URL randomly from the main dataset to ensure unbiased selection. For the purpose of evaluating our proposed solution, we developed an automated test bed for testing the tool using java. We also assume that the user has to key in the URI into the automated application before browsing the website to avoid any scripting hooks.

In order to measure the tool's effectiveness in steady state, we evaluated the tool over the course of two weeks of random browsing activities from the provided dataset. Table 2, summarizes the results sent by users who used the proposed tool for at least two weeks.

Table 2. High-Level Statistics of users browsing the dataset URL for at least two weeks.

Number of Users	30
Average number of Pages Browsed per day	1320
Total number of Pages Browsed in 2 week	26416

When we tested the tool with the phishtank dataset from dec 2009 till jan 2012 our tool was able to identify 96% of the phishing sites with 4% False positives.

The false positives were due to the fact that our proposed tool relies on RDF tags. However, if a website contains no RDF tags, the tool will extract all the embedded links base URLs as well as the page URL itself. The tool will then compare all the embedded base URLs with the page base URL, if no match, the tool will trigger a warning suspecting the website to be a phish website.

Possible ways to bypass Scripting hook

As long as the web page that the user is viewing is pure HTML, it is easy to mitigate phishing attacks. This is because sensitive information in the page can only be stolen after the page is

[8] `http://www.phishtank.com.`

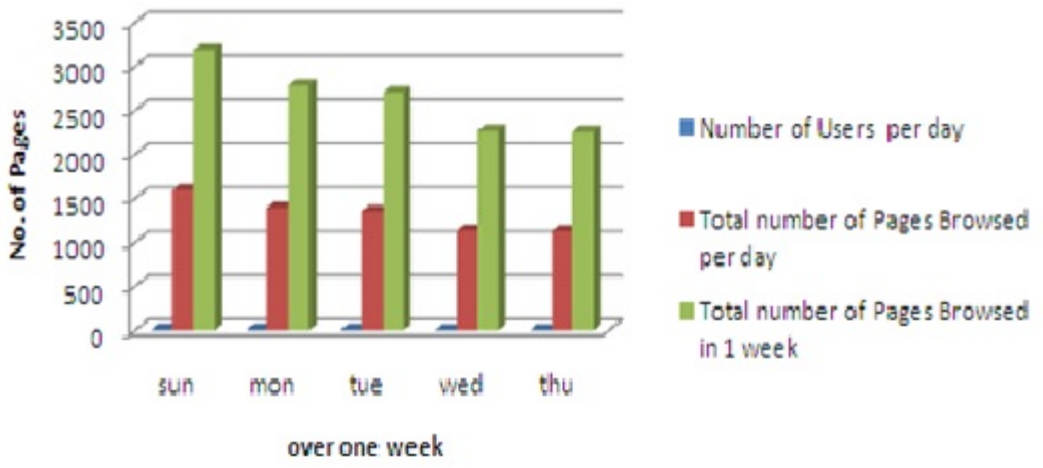

Figure 5. Total Number of pages browsed per user per day over one week.

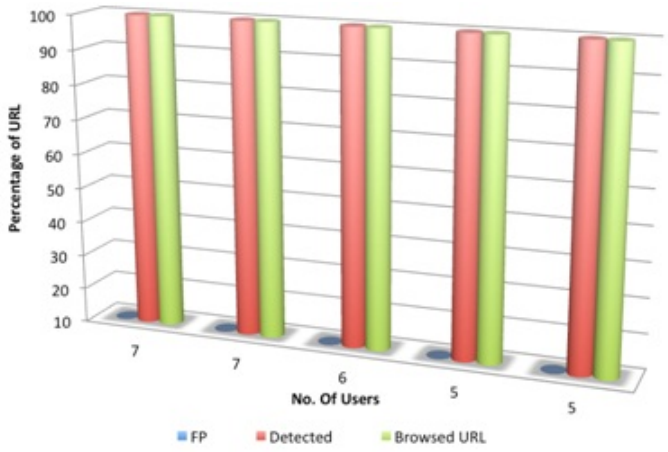

Figure 6. Detection Accuracy.

submitted. Maintaining a database of known websites can minimize the risk of phishing websites. On the other hand, stopping a phishing attacks in an HTML page that has JavaScript, is more challenging because JavaScript gives attackers a wide range of possibilities for bypassing a monitoring application. For instance, JavaScript creates hooks for capturinguser-generated events such as keystrokesembedded into the HTML page. Instead of waiting for the user to press the submit button to send the information, the attacker could intercept the keys that are pressed and send the information character by character to a server for his/her choice. Similar as setting up a phishing site at the IP address 128.131.172.93 and send out thousands of emails containing a link to this server. Cloning a bank website is trivial. The victim believes that the email is authentic and clicks a link that has been masked as: https://online.iiabank.com.jo/ConnectPres/ . When the victim starts typing his user ID and password, upon submitting the page, all the information's are sent to the attacker server. For this purpose we created many phishing websites at different local IP addresses for the testing purposes. We did not add these new phishing websites to our database to validate the feature extraction. One of the websites we are cloning is the login page of the Islamic International

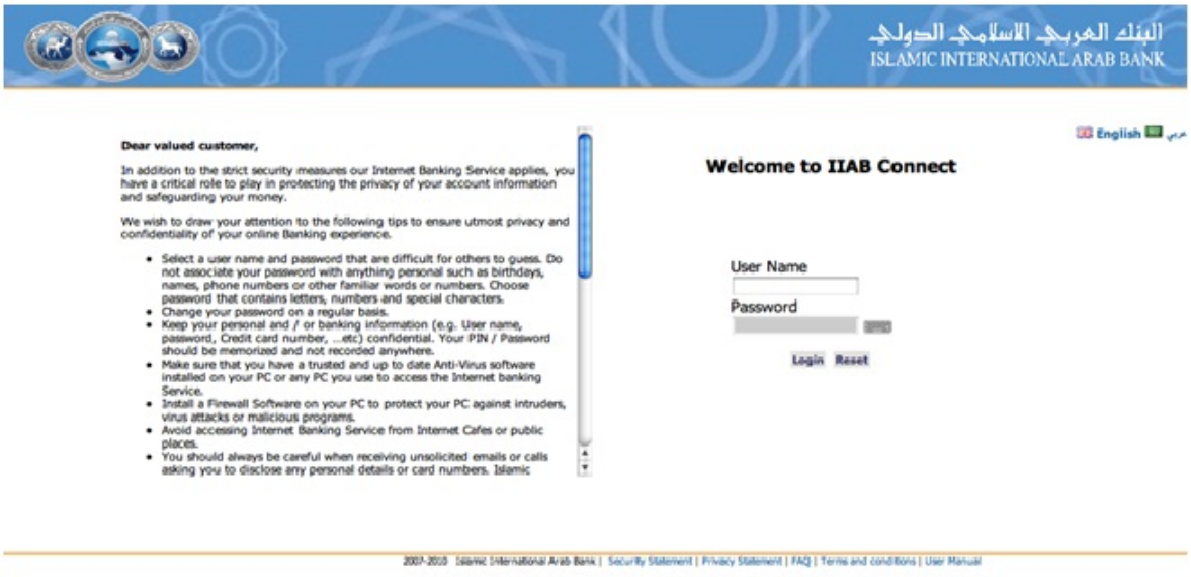

Figure 7. Islamic International Arab Bank login Page.

Arab Bank[9] as portrayed in Figure 7.

The above page contains various HTML tags that appears as follows:

```
<script type="text/javascript" src="js/login.js"></script>
<imgsrc="files_iiab/assets/common/langIcons/eng.gif" /></a>
<imgsrc="files_iiab/assets/common/langIcons/arb.gif" />
```

Note that the base URL https://online.iiabank.com.jo/ConnectPres does not appear in each HTML tag. However, these tags will be translated into the following links upon requesting the page itself.

```
https://online.iiabank.com.jo/ConnectPres//js/login.js
https://online.iiabank.com.jo/ConnectPres//files_iiab/assets/common/langIcons/eng.gif
https://online.iiabank.com.jo/ConnectPres//files_iiab/assets/common/langIcons/arb.gif
```

Where, the above links will be translated into different links pointing to the fake servers.

```
https://IP//js/login.js
https://IP//files_iiab/assets/common/langIcons/eng.gif
https://IP/files_iiab/assets/common/langIcons/arb.gif
```

The Keyword IP refers to the different local IP addresses. Note that the tags have to have the actual base URL or a local IP address pointing to the fake server. For this test, five users browse 10 various websites (i.e. cloned sites, or phishing sites) and the obtained result are portrayed in Figure 8.

The above results show a high detection accuracy that reaches 96% with false positive rate up to 4%.

[9] https://online.iiabank.com.jo/ConnectPres/

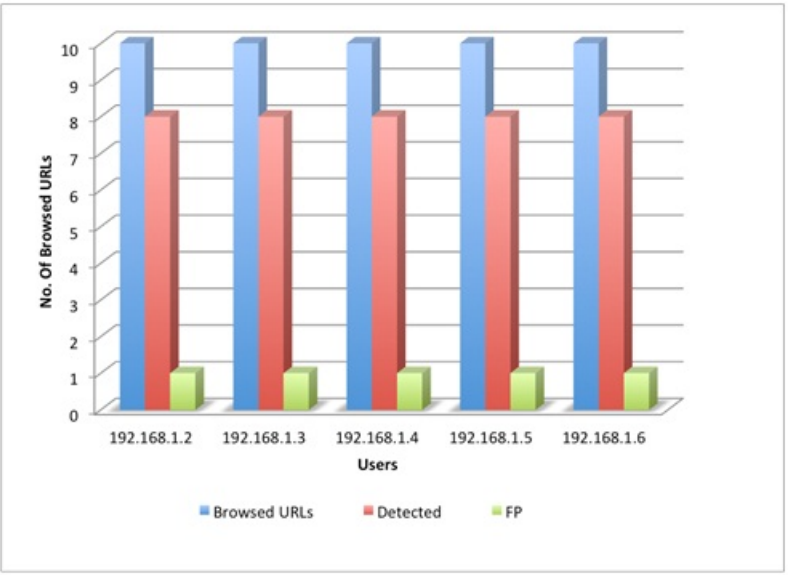

Figure 8. Detection accuracy: No Scripting Hooks.

6. Related Work

There is no standard definition for phishing attacks. However, most definitions agree that the phishing is a way of attempting to acquire individuals' personal confidential information such as usernames, passwords, and credit card details by directing users to enter details at fake websites whose look and feel is very much alike to the legitimate one [6, 7]. Phishing attacks varies in their attack methods, some attacks misdirect the user to a fake website through the Domain name System (DNS) by acquiring the domain name for certain websites and redirect their traffic to the desired phishing website (i.e Pharming) [8]. Others utilize the power of Botnets; through instructing the bots to send out phishing spam emails to a group of user's emails in a try to deceive them to follow certain embedded and/or hidden links within the spam email body; such as Asprox and kneber botnet [9].

Eventhough there are several solutions proposed and implemented for detecting and preventing phishing attacks, some techniques attempt to prevent phishing mails from being delivered [10, 11], others blacklist malicious URLs [12], and yet others analyze web pages that a user visits [13]. However, most of them suffer from unacceptable levels of false positives or negatives detection rates. This conclusion comes after the Anti-Phishing Working Group (APWG) report a total number of 29930 unique phishing reports in January 2007. This was the highest number of reports recorded by the APWG [6]. A careful investigation of the proposed solutions reveals that the focus of these works mainly depend on identifying phishing attempts through identifying certain features within the vehicle of the phishing itself (i.e spam email) [14]. For instance Fette [15], proposed a method called PILFER that depends on features extraction to distinguish between phishing email and ham email with 10 features denoted to phishing email for training data. This method has 96% accuracy with false positive rate 0.1% and 4% false negative. In [16], Abu-Nimeh compared six classifiers related to the machine learning technique for phishing prediction. He used 43 features for training and testing by six classifiers. Abu-Nimeh concludes that there is no standard classifier for phishing prediction

because if some classifiers have low levels of FP (reaching up to 04.89%,), they will have a high level of FN (reaching up to 17.04%.). Similarly with Saberi [17] who proposed a new mechanism using three learning methods for phishing e-mail detection. The mechanism depends on binary classification which is either scam or non-scam. Saberi's proposed method detected 94.4% of phishing e-mails accurately, with the FP reaching up to 0.08%. Islam [18] used another feature-based approach, which depends on three-tier classification method system to detect phishing e-mail. This technique proves that the Bayesian algorithm provide the best level of average accuracy, reaching up to 97% [19, 20].

Anti-phishing toolbars are far and wide available and commonly used by naive or non-technical computer users to help pinpointing the phishing websites such as Spoofguard [13] and Netcraft [21] toolbars as reported in [22]. However, many research studies have verified the futility of such techniques. One of the major problems with such solution is that, quite often the spoofed link is tested without any consideration to the context in which it was presented to the user, thereby losing accuracy. Another problem is that once the user enters the address of the phishing site in the browser address bar, the user is exposed immediately to any attack carried by the site [15, 23, 22].

AntiPhish is a Firefox anti-phishing browser plug-in developed in 2005 [24], It keeps track of a user's sensitive information (e.g., a password) through binding this information of a user (e.g., her password) to domain names, thus, preventing this information from being passed to a web site that is not considered trusted (or safe). The antiPhish is similar to PwdHash [25] and SpoofGuard [13], where both solutions convert a user's password into a domain-specific password. Unfortunately, the proposed solution only works for passwords but it cannot protect sensitive information that is needed in unaltered form (e.g., credit card information or social security numbers). However, the above solutions can be evaded if the attacker employs/embeds a malicious JavaScript into his phishing page. Through which the attacker has a wide range of methods to bypass any monitoring plug-ins such as AntiPhish or SpoofGuard. For instance, the attacker can hook a JavaScript to listen for key-press events rather than waiting the user to submit the form, the attacker could intercept each key that is pressed and send the information character by character back to his server. Thus, before the plug-in can detect that sensitive information is entered, most of the data is already transmitted to the attacker.

It is obvious that JavaScript can be disabled, however, disabling the JavaScript is not feasible because many web sites make use of JavaScript to submit forms and often used for client-side input validation purposes. As a result, AntiPhish author propose to deactivate JavaScript every time a form element get focus and reactivate it whenever the focus is lost in a hope to ensure that an attacker cannot capture key strokes or launch timing attacks before AntiPhish can examine any input for sensitive data. However, attackers methods getting more sophisticated daily, and new methods are born to defeat such mechanisms such as a embedding a JavaScript code that read the text in the form after it lose focus or prior of the page submission. Moreover, JavaScript cannot be deactivated and activated on demand for all web browsers, they have to be turn off on a system-wide basis [24].

7. Conclusion

Phishing is a form of online identity theft that aims to steal sensitive information from users such as online banking passwords and credit card information. Although phishing scams have received extensive press coverage, phishing attacks are still successful because of many

naïve users. Attackers are employing a large number of technical spoofing tricks such as URL obfuscation to make a phishing website look authentic to the victims.

It is believed that the best way to minimize the threat of phishing attacks is to conduct awareness programs and training sessions to users in order not to follow email's embedded links and websites unsightly. Nevertheless, not everyone is able to understand the threat of phishing. Therefore, a solution for such threat is imperative.

This article proposes to annotate an HTML document with RDF tags that aims to protect users against spoofed web site-based phishing attacks. The proposed method and tool will check the knowledge base represented as RDF for known and authentic websites. As the number of phishing scams continues to grow and the costs of the resulting damages increases, we believe that the proposed tool is a step in the right direction and a useful contribution for protecting users against spoofed web site-based phishing attacks. The proposed approach falls in the category of white-listing techniques, for which the RDF knowledge base is guaranteed since the number of legitimate banking web sites is not too huge. Moreover, each country can store an RDF knowledge base containing legitimate banking web sites. In this case, an approach for federated SPARQL queries can be adopted [5].

References

[1] G. Aaron and R. Rasmussen, "Global phishing survey: Trends and domain name use in 2h2010," 2010.

[2] Frank Manola and Eric Miller, "RDF primer," Recommendation, W3C, 2004, `http://www.w3.org/TR/rdf-primer/`.

[3] Eric Prud'hommeaux and Andy Seaborne, "SPARQL query language for RDF," Recommendation, W3C, January 2008, `http://www.w3.org/TR/rdf-sparql-query/`.

[4] Dan Brickley and R.V. Guha, "RDF vocabulary description language 1.0: RDF schema," Recommendation, W3C, 2004, `http://www.w3.org/TR/rdf-schema/`.

[5] Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt, "Fedx: A federation layer for distributed query processing on linked open data," in *ESWC (2)*, 2011, pp. 481–486.

[6] "Anti-phishing working group. http://www.antiphishing.org/.anti-phishing working group. http://www.antiphishing.org/.," .

[7] A. Emigh, "Online identity theft: phishing technology, chokepoints," Tech. Rep., and countermeasures. Technical Report, Infosec Technology Transition Council, Department of Homeland Security, 2005. Accessed at http://www. cyber. st. dhs. gov/docs/phishing-dhs-report. pdf, 2005.

[8] "An introduction to rdf and the jena rdf api. "http://jena.sourceforge.net/tutorial/RDF_API/index.html"," .

[9] Y. Shin, S. Myers, and M. Gupta, "A case study on asprox infection dynamics," *Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 1–20, 2009.

[10] "Microsoft. sender id framework overview. http://www.microsoft.com, 2005," .

[11] "Yahoo. yahoo! anti-spam resource center. http://antispam.yahoo.com, 2006.," .

[12] "Microsoft. anti-phishing technologies. http://www.microsoft.com, 2005.," .

[13] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J.C. Mitchell, "Client-side defense against web-based identity theft," in *Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSSŠ04)*, 2004.

[14] M. Chandrasekaran, K. Narayanan, and S. Upadhyaya, "Phishing email detection based on structural properties," in *NYS Cyber Security Conference*, 2006.

[15] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 649–656.

[16] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, "A comparison of machine learning techniques for phishing detection," in *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*. ACM, 2007, pp. 60–69.

[17] A. Saberi, M. Vahidi, and B.M. Bidgoli, "Learn to detect phishing scams using learning and ensemble? methods," in *Web Intelligence and Intelligent Agent Technology Workshops, 2007 IEEE/WIC/ACM International Conferences on*. IEEE, 2007, pp. 311–314.

[18] M.R. Islam, J. Abawajy, and M. Warren, "Multi-tier phishing email classification with an impact of classifier rescheduling," in *Pervasive Systems, Algorithms, and Networks (ISPAN), 2009 10th International Symposium on*. IEEE, 2009, pp. 789–793.

[19] J. Yearwood, M. Mammadov, and A. Banerjee, "Profiling phishing emails based on hyperlink information," in *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*. IEEE, 2010, pp. 120–127.

[20] R. Dazeley, J. Yearwood, B. Kang, and A. Kelarev, "Consensus clustering and supervised classification for profiling phishing emails in internet commerce security," *Knowledge Management and Acquisition for Smart Systems and Services*, pp. 235–246, 2011.

[21] "Netcraft ltd. netcraft toolbar, 2011. http://toolbar.netcraft.com/," .

[22] L. Cranor, S. Egelman, J. Hong, and Y. Zhang, "Phinding phish: An evaluation of anti-phishing toolbars," *CyLab, Carnegie Mellon University*, 2006.

[23] M. Wu, R.C. Miller, and S.L. Garfinkel, "Do security toolbars actually prevent phishing attacks?," in *Proceedings of the SIGCHI conference on Human Factors in computing systems*. ACM, 2006, pp. 601–610.

[24] T. Raffetseder, E. Kirda, and C. Kruegel, "Building anti-phishing browser plug-ins: An experience report," in *Proceedings of the Third International Workshop on Software Engineering for Secure Systems*. IEEE Computer Society, 2007, p. 6.

[25] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J.C. Mitchell, "Stronger password authentication using browser extensions," in *Proceedings of the 14th Usenix Security Symposium*, 2005, vol. 1998.

Authors

Faisal Alkhateeb is an assistant professor in the department of computer science at Yarmouk University. He holds a B.Sc. from Yarmouk University, 1999; a M.Sc. from Yarmouk University, 2003; a M.Sc. from Grenoble 1, 2004; and Ph.D. from Grenoble 1, 2008. He is interested in knowledge-based systems, knowledge representation and reasoning, constraint satisfaction and optimization problems, intelligent systems, Semantic Web, and Artificial Intelligence.

Ahmed M. Manasrah is a senior lecturer at the Faculty of Information Technology and Computer Sciences, Yarmouk University, Irbid, Jordan since 2010. Dr. Manasra served as the Deputy Director (Research and Innovation) and the Head of iNetmon project at the National Advanced IPv6 Centre (NAv6) in Universiti Sains Malaysia. Since 2009. Dr. Manasra started his career as a web developer at Telaterra LLC from 2003 until 2004. Between 2005 until 2008, Dr. Ahmed worked as senior research officer in NRG, Universiti Sains Malaysia. He is interested in network monitoring and security.

Abed Al Raoof Bsoul is an assistant professor in the department of computer science at Yarmouk University. He holds a B.Sc. from Yarmouk University, 2002; a M.Sc. from Yarmouk University, 2004; and Ph.D. from Virginia Commonwealth University, 2011. He is interested in biomedical decision support systems, intelligent systems and information retrieval.