Phishing Websites Identification Using Data Mining Techniques

[1]P.Nitin Chandra, [2]Dr.K.Madhavi

[1]B.Tech Scholar, Department of Computer Science and Engineering, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad, Telangana – 500 090.

[2]Professor, Department of Computer Science and Engineering, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad, Telangana – 500 090

Abstract- There are number of users who purchase products online and make payment through e- banking. There is e- banking websites who ask user to provide sensitive data such as username, password or credit card details etc. often for malicious reasons. This type of e-banking websites is known as phishing website. In order to detect and predict e-banking phishing website, we proposed an intelligent, flexible and effective system that is based on using classification Data mining algorithm. We implemented classification algorithm and techniques to extract the phishing data sets criteria to classify their legitimacy. The e-banking phishing website can be detected based on some important characteristics like URL and Domain Identity, and security and encryption criteria in the final phishing detection rate. Once user makes transaction through online when he makes payment through e-banking website our system will use data mining algorithm to detect whether the e-banking website is phishing website or not. This application can be used by many E-commerce enterprises in order to make the whole transaction process secure. Data mining algorithm used in this system provides better performance as compared to other traditional classifications algorithms. By using this system user can make purchase products online products securely.

I. INTRODUCTION

There are number of users who purchase products online and make payment through e- banking. There is e- banking websites who ask user to provide sensitive data such as username, password or credit card details etc. often for malicious reasons. This type of e-banking websites is known as phishing website. In order to detect and predict e-banking phishing website, we propose an intelligent, flexible and effective system. The phishing website can be detected based on some important characteristics like Rank and Website Information that help in judging the Genuineness of a website.

- The phishing mechanism is as shown in the Figure 3. The fake website is the clone of targeted genuine website, and it always contains some input fields (e.g., text box). An attacker steals the credentials of the innocent user by performing following steps:
  - Construction of Phishing Site
  - URL Sending
  - Stealing of the Credentials
  - Identity Theft

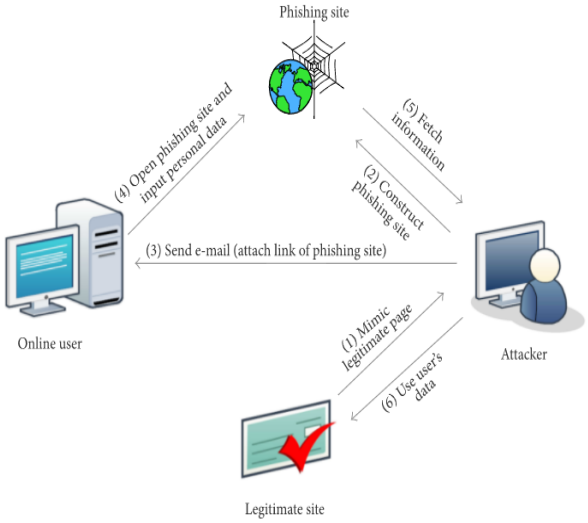


Fig 1: Phishing Mechanism

Types of Phishing Attacks:

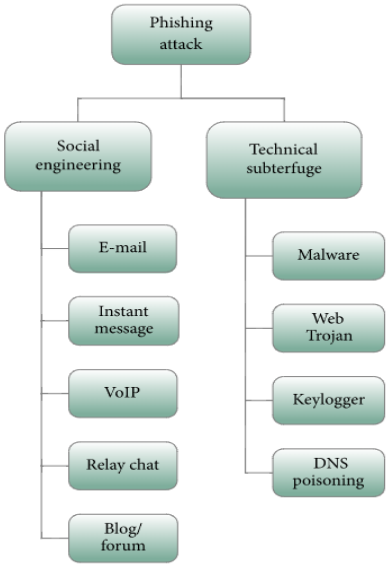


Fig 2: Phishing Attacks

Problem Statement:

- To identify the phishing websites.
- To assist the user in safeguarding their credentials and in judging the genuineness of a website.

II. EXISTING SYSTEM

Visual Similarity Assessment:

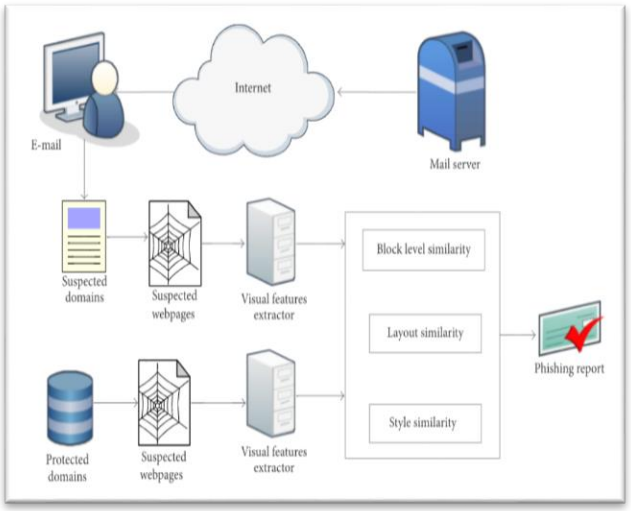


Fig 3: Current Phishing Identification

Site Signatures:

- It is a technique which creates unique web-based signature to identify the legitimate websites.
- When a user opens a new webpage, the system matches the signature of presently opened webpage with stored signature in the database.
- If the signature matches but the domain name is different, then the webpage is declared phishing.
- The proposed technique initially creates signature for the newly visited website and matches it with stored signature.

PhishZoo:

- It is an anti-phishing solution which creates the unique profile for a website using URL, text contents, images (specially website logo), Secure Socket Layer (SSL) certificate, and scripts.
- PhishZoo matches the profile of the new site with the stored profiles in the database.
- PhishZoo stored the list of legitimate sites and their profiles in the profile database.

CSS Similarity:

- Cascading Style Sheets (CSS) is a language used for depicting the formatting of a document and setting the visual appearance of a webpage written in the HTML, XHTML, and XML. CSS is used to design the webpage content like fonts, colors, and page layout.
- BaitAlarm:
  It is an algorithm to compare the CSS similarity between suspicious and legitimate website.

III. PROPOSED SYSTEM

- We have created an anti-phishing tool that will pass the URL through some condition and will check whether the URL is legitimate or a phishing website.
- We use Alexa API to fetch some of the parameters.
- The data.alexa.com is the website is used to which the URL of the website is entered and the website rank can be fetched.
- The WhoIs.com website is used to which the URL is also entered, and the registration and age of the website can be calculated. If the website's age is less than 6 months, then it has a maximum probability of being a phishing website.
- In this way, these two additional websites help in identifying the true nature of the specified website.

IV. IMPLEMENTATION

Servlet and JSP technology:

Servlet and JSP technology has become the technology of choice for developing online stores, interactive

A Servlet's Job:

Servlets are Java programs that run on Web or application servers, acting as a middle layer between requests coming from Web browsers or other HTTP clients and databases or applications on the HTTP server. Their job is to perform the following tasks, as illustrated in Figure 1–1.

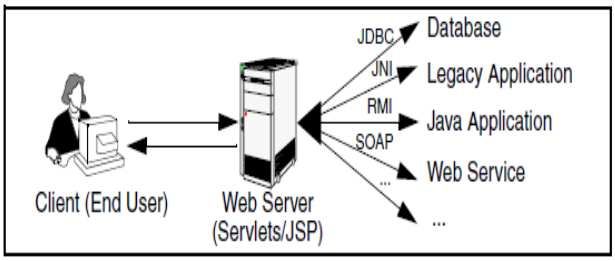


Figure 1–1 The role of Web middleware.

1. Read the explicit data sent by the client.

The end user normally enters this data in an HTML form on a Web page. However, the data could also come from an applet or a custom HTTP client program. Chapter 4 discusses how servlets read this data.

2. Read the implicit HTTP request data sent by the browser.

Figure 1–1 shows a single arrow going from the client to the Web server (the layer where servlets and JSP execute), but there are really *two* varieties of data: the explicit data that the end user enters in a form and the behind-the-scenes HTTP information. Both varieties are critical. The HTTP information includes cookies, information about media types and compression schemes the browser understands,

3. Generate the results.
This process may require talking to a database, executing an RMI or EJB call, invoking a Web service, or computing the response directly. Your real data may be in a relational database. Fine. But your database probably doesn't speak HTTP or return results in HTML, so the Web browser can't talk directly to the database. Even if it could, for security reasons, you probably would not want it to. The same argument applies to most other applications. You need the Web middle layer to extract the incoming data from the HTTP stream, talk to the application, and embed the results inside a document.
4. Send the explicit data (i.e., the document) to the client.
This document can be sent in a variety of formats, including text (HTML or XML), binary (GIF images), or even a compressed format like gzip that is layered on top of some other underlying format. But, HTML is by far the most common format, so an important servlet/JSP
task is to wrap the results inside of HTML.
5. Send the implicit HTTP response data.
Figure 1–1 shows a single arrow going from the Web middle layer (the servlet or JSP page) to the client. But, there are really *two* varieties of data sent: the document itself and the behind-the-scenes HTTP information. Again, both varieties are critical to effective development. Sending HTTP response data involves telling the browser or other client what type of document is being returned (e.g., HTML), setting cookies and caching parameters

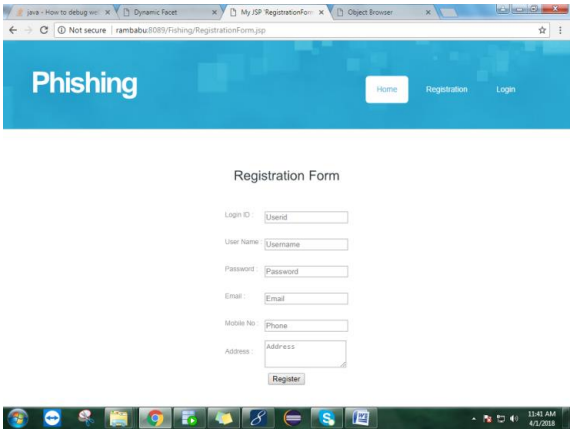
The Advantages of Servlets Over "Traditional" CGI
Java servlets are more efficient, easier to use, more powerful, more portable, safer, and cheaper than traditional CGI and many alternative CGI-like technologies. With traditional CGI, a new process is started for each HTTP request. If the CGIprogram itself is relatively short, the overhead of starting the process can dominatethe execution time. With servlets, the Java virtual machine stays running and handleseach request with a lightweight Java thread, not a heavyweight operating system process.Similarly, in traditional CGI, if there are *N* requests to the same CGI program,the code for the CGI program is loaded into memory *N* times.
Servlets have an extensive infrastructure for automatically parsing and decoding HTML form data, reading and setting HTTP headers, handling cookies, tracking sessions, and many other such high-level utilities. In CGI, you have to do much of this yourself. Besides, if you already know the Java programming language, why learn Perl too? You're already convinced that Java technology makes for more reliable and reusable code than does Visual Basic, VBScript, or C++. Why go back to those languages for server-side programming?
PowerfulServlets support several capabilities that are difficult or impossible to accomplish with regular CGI. Servlets can talk directly to the Web server, whereas regular CGI programs cannot, at least not without using a server-specific API. Communicating with the Web server makes it easier to translate relative URLs into concrete path names, for instance. Multiple servlets can also share data, making it easy to implement database connection pooling and similar resource-sharing optimizations. Servlets can also maintain information from request to request, simplifying techniques like session tracking and caching of previous computations.

The Role of JSP:
A somewhat oversimplified view of servlets is that they are Java programs with HTML embedded inside of them. A somewhat oversimplified view of JSP documents is that they are HTML pages with Java code embedded inside of them. For example, compare the sample servlet shown earlier (Listing 1.1) with the JSP page shown below (Listing 1.2). They look totally different; the first looks mostly like a regular Java class, whereas the second looks mostly like a normal HTML page. The interesting thing is that, despite the huge apparent difference, behind the scenes they are the same. In fact, a JSP document is just another way of writing a servlet. JSP pages get translated into servlets, the servlets get compiled, and it is the servlets that run at request time. So, the question is, If JSP technology and servlet technology are essentially equivalent in power, does it matter which you use? The answer is, Yes, yes, yes! The issue is not power, but convenience, ease of use, and maintainability. For example, anything you can do in the Java programming language you could do in assembly language. Does this mean that it does not matter which you use? Hardly. JSP is discussed in detail starting in Chapter 10. But, it is worthwhile mentioning now how servlets and JSP fit together. JSP is focused on simplifying the creation and maintenance of the HTML. Servlets are best at invoking the business logic and performing complicated operations. A quick rule of thumb is that servlets are best for tasks oriented toward *processing*, whereas JSP is best for tasks oriented toward *presentation*. For some requests, servlets are the right choice. For other requests, JSP is a better option. For still others, neither servlets alone nor JSP alone is best, and a combination of the two (see Chapter 15, "Integrating Servlets and JSP: The Model View Controller (MVC) Architecture") is best. But the point is that you need *both* servlets and JSP in your overall project: almost no project will consistentirely of servlets or entirely of JSP. You want both

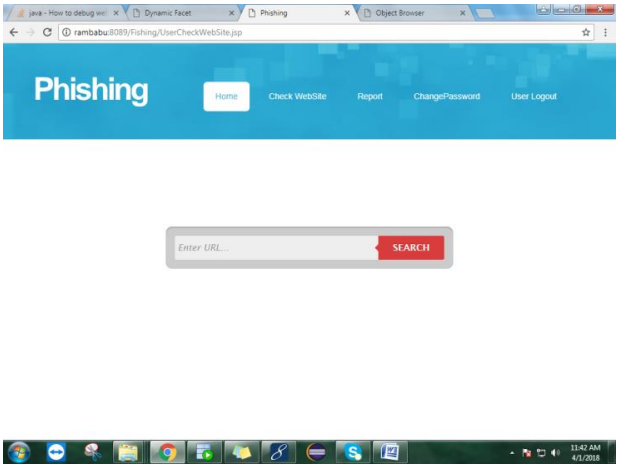
V. RESULTS

Home Page:
Website Home Page simply consists of the Home,Registration and Login tabs.

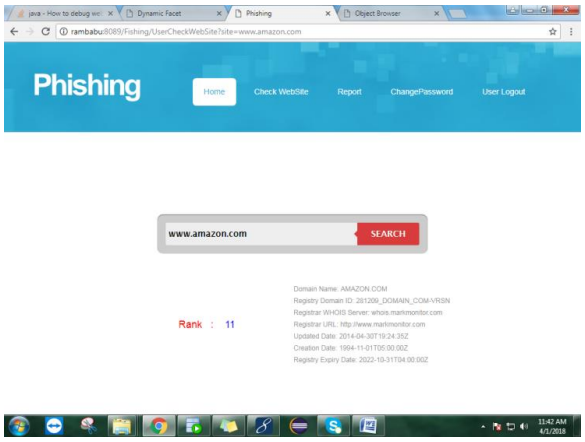
Registration Form:
The Registration Form requests for only specific and mandatory information. Notifications are sent to the user's email.



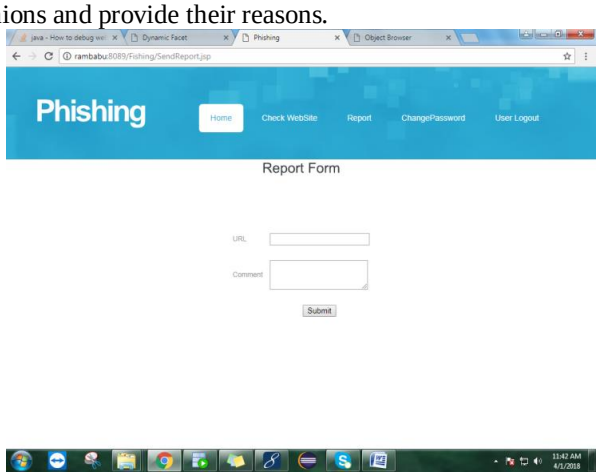
Check Website:
The Website URL is to be entered to check in the database.



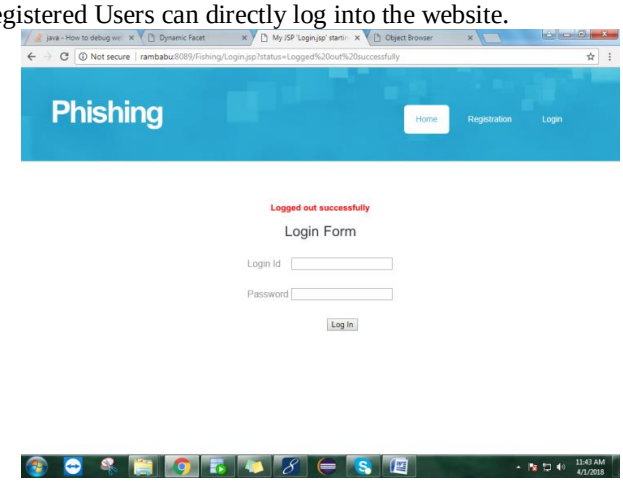
Information from Alexa API and WhoIs.com:
The URL is searched and the required data is presented to the user using Alexa API and WhoIs.com as resources.



Report Form:
Based on the presented data, the users can judge whether that particular website is genuine or not. And the users can state their opinions and provide their reasons.

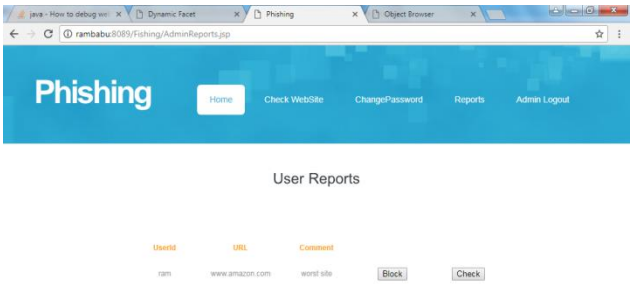


Login Form:
Registered Users can directly log into the website.



User Reports

In the Administrator account, all the comments from the users are stored and can be viewed. The administrator takes the final decision to report the website as a fake one. When a website is adjudged to be fake it will be entered into the blocked list and will be directly reported as a phishing website in the event of a future search of that website.



Based on the administrator's view of the genuineness of the website, the quality of the website is concluded. The more the website is used, the more number of various websites are updated to the administrator for his decision the website status. In this way, our project will improve with maximum usage. The blocked list will be updated with every new phishing website the administrator is aware of. Thus, our website will keep improving to help the users identify phishing websites quickly and accurately.

VI. CONCLUSION

- Phishing websites are one of the most common ways used to acquire information from users without their knowledge. Only after all the damage is done and when the money is stolen from their bank accounts will they realize that something went wrong. In this way, our website provides a simple and non-complex solution in identifying these websites.
- All the users need to do is to check the website information properly before the taking a next step. Any website, provided that it is an actual existing website irrespective of its genuineness, will have a certain amount of information that can be used to in judging its true nature.
- Thus, this project will serve as an easier solution to a very complex problem. Many more advancements can be made, and these types of solutions can be carried forward to completely erase the problem of phishing in the online world.

VII. REFERENCES

[1] Herbert Schildt,Java The Complete Reference,2017
[2] George Reese,Database Programming with JDBC and Java,1997
[3] Brett McLaughlin,Java and XML,2000
[4] P.W.Singer and Allan Friedman,Cybersecurity and Cyberwar,2013
[5] Paul Roberts,Guide to Project Management,2013
[6] Lance James,Phishing Exposed,2006