Phishing attack Generalization

Anupam Chandrayan

anupam.chandrayan@gmail.com

Abstract: Most of the attacks on web application occur by hits and trial methods, analysis of previous methods, misguiding users etc. User's are intelligent but could see only, that, which are visible. Internal configuration or working are hidden by the user, even by designer also, because web application has to cross various platforms, architectures and layers each having their own pattern or procedure, various web attacking classes are acting on web ,out of which phishing attack class has been discussed and which is the most threatening way to misguide user, Here Phishing attack has been discussed from where attacker generates exact replica of original site to get username and password.

Introduction: Phishing attack occurs by creating identical web page as like given by secure side, User puts his username and password is captured by attackers,
Example:

Like Bank site: www.Abank.com (Original Site)

www.A-bank.com(Phishing attack) Attacker confuses user, and when user accesses that fraud site, the backend script is created and username and passwords are stored at attackers database, and error message is generated, as a result bank and secure site started two way or sided scheme to provide authenticity for accessing secure connection. This Various techniques are available for detecting and eliminating suspicious and blacklisted URL's. [1] Algorithm and tool works by detecting and checking attack signatures and attack procedure and patterns. They provide bypassing mechanism to access secure connections and designs. every attack have their predefined pattern but they grows as the security implication increases, as security increases ,attack potential also increases. Every web-browser has their own deign pattern and algorithms like internet explorer, Netscape navigator, Mozilla Firefox, Opera, Google chrome, some them works on HTTP and some them works on HTTPS. Https provides secure channel for web application and are less susceptible to attack but http is susceptible to attack.

The rest of the paper describes phishing attack detection architecture and security measures for phishing attack.

Related work:

IDBC-Checker [2, 3] was not developed with the intent of detecting and preventing general SQLIAs, but can be used to prevent attacks that take advantage of type mismatches in a dynamically-generated query string. As most of the SQLIAs consist of syntactically and type correct queries so this technique would not catch more general forms of these attacks.

Wassermann and Su propose Tautology Checker [8] that uses static analysis to stop tautology attack. The important limitation of this technique is that its scope is limited to tautology and cannot detect or prevent other types of attacks.

Huang ad colleagues [6] propose WAVES, a black box technique for testing web applications for SQL injection vulnerabilities. Te tool identify all points [5] a web application that can be used to inject SQLIAs. It builds attacks that target these points and monitors the application how response to the attacks by utilizes machine learning.

Anomaly detection [8] is an approach to intrusion detection that is complementary to the use of signatures. The anomaly-based detection techniques are successful for detecting new attacks. A thread base approach is used in [2] for anomaly detection but solution is static in nature and updating the system is tedious job. Control flow graph and program slicing is explained in [1] for the validation of user input but the focus is not most critical attacks like XSS and SQL Injection at application level.

Different types of learning-based anomaly detection techniques have been proposed to analyze different data streams. Valeur et al. proposed an anomaly-based system [3] that learns the profiles of the normal database access performed by web-based applications using a number of different models. Estevez. et al. proposed an approach [7] which used Markov model to detect the web application attacks. Their approach is based on the monitoring of incoming HTTP requests to detect attacks. Naiman uses the statistic methods [6] to analyze the data which are collected from web servers However, all of these researches just considered the validation of user input. They did not consider the relationship between the page transition with the page attributes and unreasonable user visiting behavior.

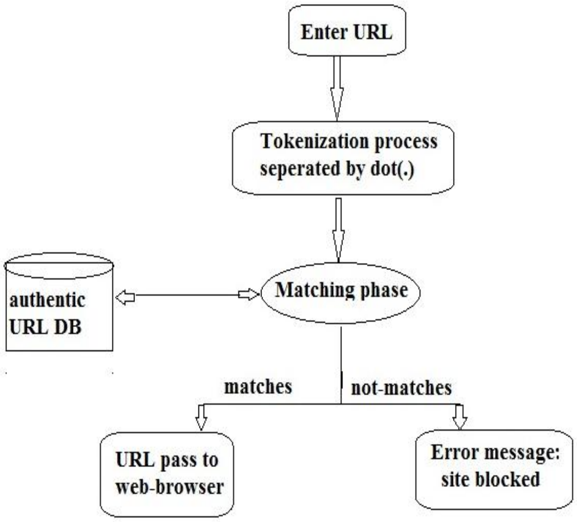
Propose work: Phishing attack is the most disastrous attack, which is made to capture user’s information entered into site.

Ex: www.abcbank.com (Original Site)

www.abc-bank.com(Attacker’s Site)

Here, attacker creates exact same structure as that of original one. User are unknown by the design, enters his username and password by seeing same atmosphere like original one, Once attacker gets user information, he is free to create vulnerable event on user's account. Below is the architecture shown in Fig 1 .

Fig 1.System for generalization



Here a strong architecture has been present for protecting or creating safe URL, The tokenization Module creates tokens separated by dot (.)Of entered URL and the matching phase the generated tokens from the Authentic database which contains fingerprint of secure or safe URL tokens and Suspicious or malicious URL tokens, one to one matching is created based on tokenization process for generating authentic results. The authentic URL database is updated by analysis or behavior of new patterns or signatures, if found in URL field, The proposed technique is most secure and best design for eliminating Phishing attacks, as here, all the patterns are detected and eliminated and recorded for future analysis and fingerprinting behavior. If URL is free from Phishing attack Ural link is forwarded to Web-Brower for execution by Analyzer, if URL contains suspicious or any analyzed attack signature it is blocked and checked for future references.

Algorithm:

1) Enter URL.
2) Generate Tokens separated by tokens.
3) Pass Tokens to matching case, here attack finger prints are checked and analyzed by authentic DB.
4) If attack signature is found.
    a) Blocked by analyzer module
    b) Else pass to web-browser

5) Repeat step 2 to 4 for further analysis.

6) Analyze the result.

Results: The Proposed system is tested on different URL's and their results are taken, the system performance is evaluated using different parameters.

Fig.2 (Comparison of TNR and Training Time)

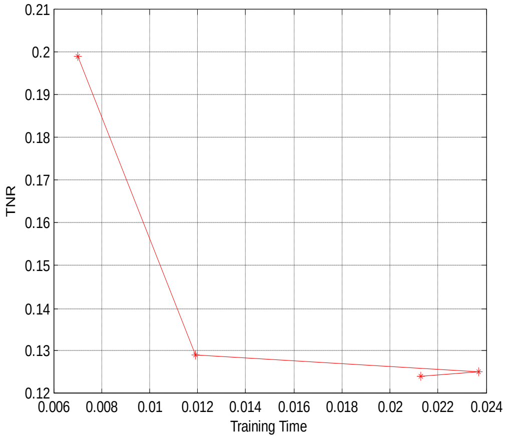


Fig.3 (Comparison of TPR and Matching Accuracy)

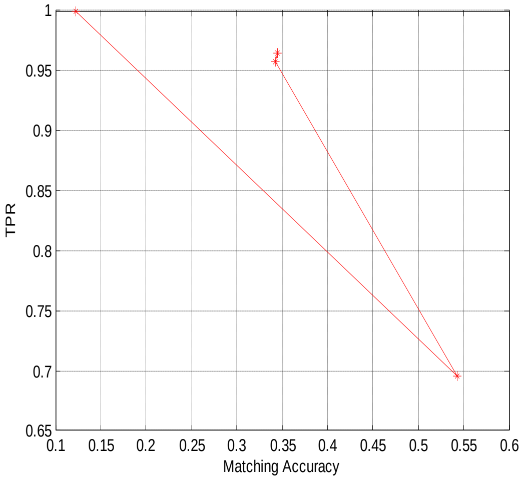
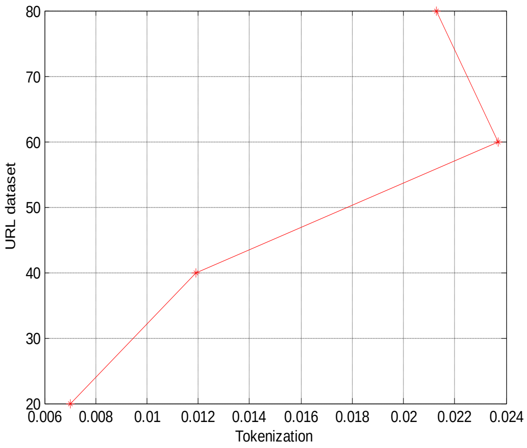


Fig.4 (Comparison of URL dataset and Tokenization)



Conclusion: The technique provides best results and system behavior as it is easy to implement. And it does not put system overhead. From the graph shown it is found that, the matching accuracy is 95%.

References:

[1]Abdul Razzaq, Ali Hur, Nasir Haider, Farooq Ahmad, "Multi-Layered Defense against Web Application Attacks", IEEE 2009.

[2]Atefeh Tajpour, Maslin Massrum ,"Comparison of SQL Injection Detection and Prevention Techniques", IEEE 2010.

[3]Yu-Chin Cheng ,Chi-Sung Laih ,Gu-Hsin Lai ,Chia-Mei Chen ,Tsuhan Chen,"Defending On-Line Web Application Security with User-Behavior Surveillance",IEEE 2008.

[4]. B. W. W. G. T. Buehrer and P. A. G. Sivilotti, “Using parse tree validation to prevent sql injection attacks,” in International Workshop on Software Engineering and Middleware, Lisbon, Portugal, September 2005.

[5]. William G.J. Halfond, Alessandro Orso and Panagiotis Manolios. Using Positive Tainting and Syntax-Aware Evaluation to Counter SQL Injection Attacks. SIGSOFT’06/FSE-14, November 5-11, 2006, Portland, Oregon, USA.

[6]. Tadeusz Pietraszek and Dhris Vanden Berghe. Defending against Injection Attacks through Context-Sensitive String Evaluation. Proceedings of Recent Advances in Intrusion Detection (RAID2005).

[7]. Finding Application Errors and Security Flaws Using PQL: a Program Query Language. OPSLA’05, October 16-20, 2005, San Diego, California, USA.

[8]. Z.Su and G. Wassermann. The Essence of Command Injection Attacks in Web Application. In the 33rd Annual Symposium on Principles of Programming languages, pages 372-382, Jan. 2009.