Arab American University – Jenin

Faculty of Graduate Studies

Phishing analysis and developing anti-phishing techniques.

By

Abdelmunem Ismail Thiab Abuhasan

Supervisor

Dr. Adwan Yasin

This thesis was submitted in partial fulfillment of the requirements for the Master`s degree in

Computer Science

January/ 2017

Phishing analysis and developing anti-phishing techniques.

By

Abdelmunem Ismail Thiab Abuhasan

This thesis was defended successfully on 22/1/2017 and approved by:

Committee members	Signature
1. Supervisor Name: Dr. Adwan Yasin	.....
2. Internal Examiner Name: Dr. Mohammad Hamarsheh	.....
3. External Examiner Name: Dr. Hani Salah	.....

Declaration

This is to declare that the thesis entitled "Phishing analysis and developing anti-phishing techniques" under the supervision of Dr. Adwan Yasin is my own work and does not contain any unacknowledged work or material previously published or written by another person, except where due reference is made in the text of the document.

الإهداء

أهدي هذا العمل و النجاح الى والدي الغاليين أمد الله في عمرهما

و إلى من اقتطعتُ من وقتهم لاستكمال دراستي، زوجتي و ابنائي

و إلى كل من شجعني و ساعدني على إتمام هذه الرسالة.

الشكر و التقدير

أشكر الله تعالى أن حقق لي ما أصبو إليه في استكمال درجة الماجستير في علم الحاسوب، كما أشكر الجامعة العربية الأمريكية على أن هيأت لي الظروف لاستكمال دراستي و قدمت لي كل الدعم و المساندة، فللقائمين عليها مني جزيل الشكر و العرفان.

و أتقدم بعظيم الشكر و الامتنان من استاذي الدكتور عدوان ياسين على حسن رعايته لهذه الرسالة و دعمه و صبره اللامحدودين، كما أشكر جميع طاقم التدريس في كلية الدراسات العليا في الجامعة العربية الأمريكية و الذين كان لهم الأثر الكبير في توجيهي و امدادي بالمعارف و العلوم التي أنارت لي الدرب لاستكمال هذه الدراسة.

وأتقدم بجزيل شكري أيضا لأساتذتي الأفاضل:

الدكتور محمد حمارشة مناقشاً داخلياً

الدكتور هاني صلاح مناقشاً خارجياً

على تفضلهما بمناقشة هذه الرسالة.

Abstract

Phishing is a kind of internet fraud that employs socially engineered messages to deceive users into declaring their sensitive information -including their credentials. Phishing attacks start by communicating with a user or a group of users using a professional email message phone call, SMS, or any other electronic method that draw an illusion to the user that it comes from a legitimate source as an institution in which the user has an account. This message requests the user to declare his credentials by submitting them to a fake website that is professionally designed to be similar to the original website of the institution. This type of electronic attacks has been adapting their nature to the countermeasures that are implemented by web site vendors and users, and became a real threat to financial institutions and electronic commerce sites.

The proposed work will focus on fighting phishing attacks using two strategies. The first strategy will focus on preventing phishing attacks by solving the root causes or weakness points in current web authentication schemes. In this context, we propose two novel authentication schemes that are immune to phishing attacks. The first one extends the authentication process into a new level that leverages the user's mobile phone as a second authentication factor; the user through a dedicated mobile application shall confirm every login attempt. This scheme depends on the ubiquitous nature of modern smart phones and internet connectivity and employs Google Cloud Messaging service for sending login notifications the user's mobile application. The second authentication scheme addresses the weakness of password-based authentication. The proposed scheme replaces password-based authentication with a new authentication strategy that leverages the user's mobile as an

identity prover. The proposed scheme applies mutual authentication between the user's mobile and the website using digital signatures and a symmetric shared key. When the user initiates a login request, the server will respond by an encrypted login token encapsulated in a QR code, the code will be processed by the user's mobile application that presents the user identity to the server.

The second proposed strategy in fighting phishing attacks focuses on mitigating phishing attacks by applying a smart phishing email classifier on the email system level. The proposed scheme applies the knowledge discovery model, data mining techniques and semantic text processing techniques to build an intelligent classifier that is able to classify phishing content at the early stage of the phishing campaign. The proposed classifier was tested on two accredited data sets composed of more than 10000 phishing and legitimate emails; it achieved an incredible positive classification rate of 99.1 % using the random forest algorithm.

Table of contents

List of tables

List of figures

List of abbreviations

API - Application Programming Interface

BMBAT - Bastion Mobile Based Authentication Technique

EARMAT - Enhancing Anti-phishing with a Robust Multi-level Authentication Technique

FCM – Firebase Cloud Messaging

FN - False Negative

FP - False Positive

GCM - Google Cloud Messaging

IMEI - International Mobile Equipment Identity

JSF - Java Server Faces

LID - Login Attempt ID

MOTP - Mobile One-Time Password

MPR - Mobile Private Key

MPU - Mobile Public Key

NLR - Negative Login Result

NMSG - Notification Message

OTP - One Time Password

PWD - User password

QR - Quick Response

SOTP - Server One-Time Password

SPR - Server Private Key

SPU - Server Public Key

SSL – Secure Socket Layer

TF–IDF - Term Frequency-Inverse Document Frequency

TLS – Transport Layer Security

TN - True Negative

TP - True Positive

1 Introduction

The internet evolution attracted most business institutions to provide their transactions online through web-based applications, among them, banks, stocks and ecommerce websites are widely spread nowadays. As people increasingly rely on Internet to do business, Internet fraud becomes a great threat to people's privacy and safety of their web-based transactions. Internet fraud uses misleading socially engineered messages to deceive human users into forming a wrong belief and take dangerous actions to compromise their or other people's private information.

The main type of Internet fraud is phishing; which relies on fooling users to share or declare their private information (like passwords and credit card numbers). Phishing attacks could be defined as a computer attack that communicates socially-engineered messages to humans through electronic communication channels (like email, SMS, phone call) in order to persuade them to do certain actions (like entering credentials, credit card number or any other confidential information) for the attackers benefit. Such actions could be persuading an e-commerce web site user to enter his credentials to a fake web site managed by the attacker similar to the original website. Then the attacker uses theses information to impersonate the user. In order to persuade the user to login to such a fake website the attack should create a need for the end-user to perform such action, such as informing him that his/her account would be suspended unless he logs in to update certain pieces of information (X. DONG et al., 2008).

Phishing attacks use emails and websites designed to look like emails and websites from legitimate institutions and organizations (user under attack is a customer for those organizations), to deceive users into disclosing their personal or financial information. The phisher can then use this information for criminal purposes, masquerading and fraud.

Users can be tricked into disclosing their information either by providing sensitive information via a web form, replying to spoofed emails, or downloading and installing Trojans, which search users' computers or monitor users' online activities in order to get information.

1.1 Importance of subject

Phishing attacks have steadily increased to match the growth of electronic commerce, recently taking on epidemic proportions. According to Anti Phishing Work Group (APWG) report of 2015 (APWG REPORT, 2015), the total number of unique phishing sites detected from Quarter1 through Quarter3 of 2015 was 630,494, while The number of unique phishing reports submitted to APWG from Q1 through Q3 was 1,033,698 as shown in figures 1 and 2. According to a recent Google study (GOOGLE STUDY, 2014), some fake websites succeeded 45% of the time to deceive victims to release their credentials, and around 20% of hijacked accounts are accessed within 30 minutes of a hacker obtaining the login info. Once they have broken into an account they want to exploit the information, hijackers usually spend more than 20 minutes inside, they often change the password to lock out the true owner and search for other account details like bank, or social media accounts.

In addition, attackers send phishing emails from the victim's account to everyone in his or her address book. Since the victim's friends and family trust the victim's email, these emails can be very effective. People in the contact list of hijacked accounts are 36 times more likely to be hijacked themselves (GOOGLE STUDY, 2014).

Phishing is a particularly insidious problem for institutions that offer electronic services like financial institutions, since trust forms the foundation for customer relationships, and phishing attacks undermine confidence in an institution.

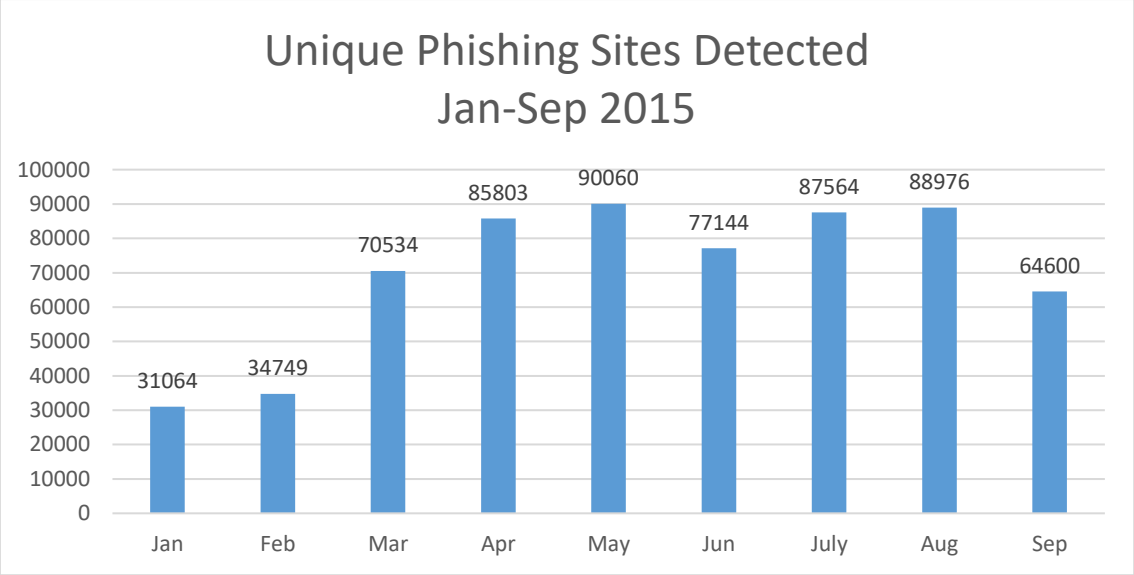

Figure 1: Unique Phishing Sites Detected in Q1-Q3 of 2015 (APWG REPORT, 2015)

Figure 2: Phishing reports submitted by the community to APWG in Q1-Q3 of 2015 (APWG REPORT, 2015)

1.2 Problem Statement

Phishing is one of the most popular cyber-attacks that threaten people privacy and sensitive information; in addition, it threatens the overall security of the web sites. Estimations reported a huge number of phishing attacks that are directed randomly or to specific users (Spear phishing), resulting in financial loss of millions of dollars, according

to (CLOUDMARK, 2016) the average cost of a spear phishing attack is 1.6 million dollars.

Several methods have been proposed to combat phishing attacks, including user training about phishing attacks, SSL/TLS (Secure Socket Layer / Transport Layer Security) and third-party certification, classifier and detector tools and enhanced web authentication schemes. However, none of those solutions was able to defeat the continuous and adaptable phishing threats, which raises the need for enhancing those proposed solutions and proposing different solutions to enhance the effectiveness of fighting phishing attacks.

1.3 Thesis Objectives

This thesis aims at developing solutions that contribute to the efforts of mitigating and preventing phishing attacks, the proposed solutions contribute in three directions:

- Enhancing web authentication by proposing an enhanced web authentication technique that supplements the traditional text-password based authentication with a new level that allows the user to confirm any login attempt from his mobile device. This technique protects the user account from being compromised in case the phishers were able to compromise his account credentials.
- Proposing an authentication technique that replaces the traditional text-password based authentication, this technique identifies the user through a dedicated mobile application that will act as an identity proofer using a set of cryptographic algorithms that apply the principle of mutual authentication between the user and the web server. This technique will contribute to the efforts of fighting phishing attacks through neutralizing the threat of compromising the user credentials.

- Developing a classification model for identifying phishing content as an early step in the process of fighting phishing attacks, this model utilizes data mining and text preprocessing techniques to build a smart content classifier to classify any suspicious content (emails, web pages, etc...) as legitimate or phishing content.

1.4 Organization of the thesis

Protecting user's data and privacy is one of the key success issues that a website vendor needs to guarantee to his clients, especially for those vendors whose business model relies on or includes offering online services. We believe that it is the website's owners' responsibility to prevent possible data breaches and to combat phishing attacks, they cannot rely on user's vigilance to protect their users' accounts as we will explain in chapter 2. We believe that a robust and secure authentication scheme will protect the user's account from fraud even if his password was compromised by a phisher.

In this work, the anatomy of the phishing epidemic is tackled to better understand and analyze the possible attack vectors. In addition to analyzing the current web authentication techniques and their weak points that make it possible for an attacker to impersonate a victim user. This thesis contributes to the efforts of combating phishing attacks in two directions. The first contribution introduces two enhanced authentication schemes that are capable of fighting phishing attacks and protecting the user's data in case he has been a victim of a phishing attack. The second contribution employs knowledge discovery and data mining techniques in building an intelligent email classifier that is capable of classifying phishing from legitimate email messages. The proposed work in this thesis could be summarized as follows:

1.4.1 Enhancing Anti-phishing by multi-level authentication technique (EARMAT)

This authentication scheme enhances the traditional password-based authentication scheme by introducing a new level in the authentication process. The proposed model exploits the user's mobile device to confirm/reject any authentication attempt to the user's account after mutually authenticating the web server and mobile client to each other using the PKI.

In this scheme any successful attempt to login to the user account by the right user name and password will be delegated to the confirmation request from the user's mobile, this communication is carried out by sending a notification to the user's mobile through GCM Google Cloud Messaging service (GOOGLE CLOUD MESSAGING)[1]. After that, the user is introduced to the authentication request details and he decides whether to confirm or reject it.

In a website that implements this authentication scheme, it is not sufficient for a phisher to steal the user's credentials to control his account; the phisher also needs to control the victim's mobile device and the authentication app also. The EARMAT authentication scheme analysis, implementation and evaluation is introduced in chapter 4.

1.4.2 A Bastion Mobile-Based Authentication Technique to Replace Passwords (BMBAT)

This authentication scheme fights phishing attacks through neutralizing the origin causes of phishing attacks. BMBAT eliminates the need for a password to authenticate users to websites and as a result, phishing attacks will no more succeed to fool users.

In this scheme, the user is identified by his user name only by exploiting his mobile device as an identity prover. This scheme employs a dual mode of encryption using symmetric

[1] Firebase Cloud Messaging (FCM) is the new version of GCM, for more information we refer the reader to https://firebase.google.com/docs/cloud-messaging/

and asymmetric (RSA) keys to mutually identify both the client to the server and vice-versa. The user logs in with his user name, then the server responds with a QR code that encodes a session-based nonce encrypted with the server's private key after being encrypted with a shared key with the user. The user then scans the QR code by his mobile phone camera, and extracts extracting the nonce. After that, the user's mobile acts in a smart way by sending the response to the server directly, or in case the mobile is not connected to the internet, the extracted nonce is displayed on the mobile screen so that the user can use it as a OTP and complete the login process by entering it through the browser. Upon receiving the nonce, the server verifies it and authenticates the user session.

BMBAT competes to similar schemes by enabling the user to complete the login process from his mobile device in case it is not connected to the internet, by enabling the session code (login code) to be disposed to the user in his mobile screen. BMBAT authentication scheme analysis, implementation and evaluation is introduced in chapter 5.

1.4.3 An intelligent classification model for phishing email detection

This work contributes to the early stage of fighting phishing attacks. As emails are the most common way for initiating a phishing campaign, it is very important to detect the phishing content at the email level before being processed by humans. The idea is to classify or filter phishing emails and take the appropriate actions to stop the phishing campaign at its very beginning. In this context, we introduce an intelligent classification model for detecting phishing emails using knowledge discovery, data mining and text processing techniques. The proposed work builds an intelligent classification model that learns from a training data set of more than 10000 accredited messages of both legitimate and phishing content. This classification model introduces the concept of phishing terms

weighting which evaluates the weight of phishing terms in each email. The pre-processing phase is enhanced by applying text stemming and WordNet (GEORGE A. MILLER, 1995) ontology to enrich the model with word synonyms. The proposed model applied the knowledge discovery procedures using five popular classification algorithms and achieved a notable enhancement in classification accuracy. An accuracy rate of 99.1% was achieved using the Random Forest algorithm and 98.4% using J48, which is –to our knowledge- the highest accuracy rate for an accredited data set. A comparative study with similar proposed classification techniques is also introduced to evaluate the proposed classification model.

This thesis is organized as the following, chapter 2 is a review and analysis of the anatomy of phishing attacks, and current anti-phishing approaches, chapter 3 covers the current web authentication techniques. EARMAT authentication scheme analysis, implementation and evaluation is covered in chapter 4, BMBAT authentication scheme analysis, implementation and evaluation is covered in chapter 5. Chapter 6 introduces an intelligent classification model for phishing email detection.

1.5 Publications

In this section, we list our recently accepted publications in the field of fighting phishing attacks; those publications formulate the core of our contributions in this thesis.

- ADWAN YASIN AND ABDELMUNEM ABUHASAN. (In press). “Enhancing Anti-phishing by a Robust Multi-Level Authentication Technique (EARMAT)”. The International Arab Journal of Information Technology.

- ABDELMUNEM ABUHASAN AND ADWAN YASIN. (2016). “A BASTION MOBILEID-BASED AUTHENTICATION TECHNIQUE (BMBAT)”. International Journal of Network Security & Its Applications (IJNSA) Vol.8, No.6, November 2016.

- ADWAN YASIN AND ABDELMUNEM ABUHASAN. (2016). “AN INTELLIGENT CLASSIFICATION MODEL FOR PHISHING EMAIL DETECTION”. International Journal of Network Security & Its Applications (IJNSA) Vol.8, No.4, July 2016.

2 Background and Literature Review

2.1 Background

Phishing is a kind of Internet fraud that uses spoofed emails and websites as lures to fool internet users to voluntarily provide their sensitive information. Such attacks are carried out by attackers who use a bait to persuade their victims to follow spoofed links and declare or fill in their private information through spoofed web sites for the benefit of the attackers. The term "Phishing" came from the term Phone Phreaking which is the old form of hacking that was directed against telephone networks, thus the letter "f" in the term "fishing" was replaced with letters"ph".

A set of motivations stand behind performing phishing attacks by attackers, including financial and psychological benefits, (Weider D. et. al., 2008) discussed those motivations and can be summarized as follows:

1. Financial gain: attackers use stolen user's credentials to conduct online transactions to their financial benefits on behalf of the victim, such as online purchasing, money transfer… etc.
2. Identity hiding: some phishing attacks aim at exploiting stolen identities in performing criminal actions, so that those actions are performed in the name of the victim.
3. Fame and notoriety: phishers might attack victims for the sake of peer recognition.
4. Attackers can sell phished credentials, emails and identities for advertisement agencies who seek to publish ads.
5. Manufacturing industry secrets: according to Symantec Intelligence Report (SYMANTIC, 2015), 22% of attacks where spear attacks on the manufacturing industry, aiming at gaining sensitive information on the manufacturing processes.

Data suggest that some phishing attacks have convinced up to 5% of their recipients to provide sensitive information to spoofed websites (LOFTESNESS S., 2004). About two million users gave information to spoofed websites resulting in direct losses of $1.2 billion for U.S. banks and card issuers in 2003 (LITAN, 2004). We conclude from those studies that it is extremely important to prevent or at least minimize the success rates of phishing attacks to improve the overall security of a website in addition to enhancing the overall trust in web transactions in a whole.

2.2 Challenges of combating phishing attacks

Phishing attacks success in compromising users account is usually ought to that they take advantage of users lack of experience and ignorance to security warnings. In addition to the not mature enough phishing detecting strategies that are deployed and the phishers ability to adapt their techniques to overcome phishing solutions. Those factors make phishing a not easy to solve problem.

Huge efforts have been spent to combat phishing attacks by improving users' awareness and educating them to not fall as victims to phishing attacks. Side by side, huge enhancements to automated software phishing detection/prevention tools were applied.

In the literature, the proposed solutions to detect and prevent phishing attacks can be classified into four categories:

- **End user training**

  This category of anti-phishing solutions relies upon educating users about phishing attacks and equipping them with the necessary knowledge that makes them capable of identifying and dealing with suspicious phishing content, in addition to taking the appropriate actions to avoid falling as victims to phishing attacks. An experienced user is expected to be able to differentiate between

legitimate and phishing content and URLs using the knowledge he gained during the education process.

Unfortunately, studies concluded that most users are not aware of the basic security concepts and features (RACHNA DHAMIJA et al., 2006) (JULIE S. DOWNS et al., 2006), those conclusions result in rendering end user based techniques of fighting phishing attacks not effective, especially for advanced and well-prepared phishing content.

- **SSL/TLS and third-party certification**

Secure Socket Layer (SSL) and its successor Transport Layer Security (TLS) are two security protocols that employ Public Key Infrastructure (PKI) to secure data transmitted between two parties (client and server) through encryption in addition to authenticating the server through a certificate issued by a trusted third party - also known as Certificate Authority (CA) - such as VeriSign. Using this trusted certificate, the client (the web browser in this context) can verify whether the domain name in the requested URL truly belongs to the server to which the user is connecting; so the user can decide whether the web site he is visiting is a phishing web site or not. The web browser client displays a small lock icon in the address bar to indicate a secure connection after verifying the server's certificate to be from a trusted CA, this helps the user to make sure he is connecting to the right web application.

While this defense mechanism contributes to the efforts of combating phishing websites, it relies on the user's experience and knowledge of SSL and certification to decide whether the web site is a phishing one, and thus is not considered an effective mechanism to defeat phishing attacks.

- **Anti-phishing tools**

Anti-phishing tools are software components that are attached to web browsers or email programs as toolbars or add-ons/extensions. They are capable of detecting suspicious phishing content and web pages by using a set of techniques including verifying the web site certificate, consulting black and white listings, auto classification employing artificial intelligence and data mining techniques to calculate the probability of the content to be a phishing content and taking the appropriate action.

Examples of anti-phishing tools are Spoof Guard (SPOOFGUARD , 2005) and Phish Guard (P. LIKARISH et al., 2008), while these tools are designed to combat phishing web pages, their effectiveness varies from one to another according to the techniques employed in the tool; Zhang et.al. (Y. ZHANG et al., 2007) experimented 10 anti-phishing toolbars with a set of legitimate and phishing pages, their results concluded that the best of the experimented tools (Spoof Guard) result in high false positives rate, i.e. it erroneously identified a large fraction of legitimate sites as phishing.

Anti-phishing tools suffer from the following weaknesses:

1. Incorporating false positives; i.e. classifying legitimate sites as suspected phishing sites.
2. Usability problems: Including the need to install the toolbar and that, the tool may not be available in public or shared computers, which renders them useless in this case.
3. Some anti-phishing tools are browser dependent, such as Microsoft SmartScreen filter.

4. Dependence on user behavior, i.e. most of the tools notify the user of a suspected phishing page but leave an option for him to bypass this caution and continue to visit the page.

- **Extending the traditional user authentication schemes with a second factor**

In order to resist phishing attacks, many authentication schemes were proposed to boost the security of traditional password based authentication by extending it with a second factor; such that if the user's password was compromised by a phishing attack, the attacker will not be able to compromise the user account unless he controls or owns the second factor.

One form of the second authentication factors is one-time passwords, a unique code that is communicated to the user to complete the authentication process. This code may be used only one time for the current user session and then expires. The code could be communicated to the user through SMS, email or through a mobile app code generator hosted in the user mobile, as the case in Google Authenticator (GOOGLE 2 STEP VERIFICATION).

While it enhances the security and provides better resistance against phishing attacks, a set of issues need to managed when implementing a second authentication factor, including cost, availability, deployability, usability and targeted security attacks and breaches on the second factor. Those issues are discussed in chapter 3.

While those techniques could be efficient to eliminate (or more precisely decrease) the phishing attack threats, a set of challenges arise when applying those approaches:

- Novice users usually resist learning, and if they learn it is difficult to retain their knowledge and put it in practice, and thus training should be made continuous. While some studies concluded that user training is helpful (S. SHENG et al., 2010),

(P. KUMARAGURU et al., 2007), (A. ALNAJIM AND M. MUNRO, 2009), others disagree (S. GORLING, 2006), (G. GAFFNEY, 2015).

- Software solutions that classify phishing attacks render their classification result to the user and relies upon him to decide what to do (e.g. dropping the phishing email or stop visiting the phishing website); if the user ignores this notification, then the software becomes useless. In another words, it depends on the user behavior and knowledge to benefit from such detection software.
- Phishing attacks rely mostly on emails sent to the user in natural language, this make the detection process of phishing emails more complex and usually not deterministic.
- Robust authentication based websites usually introduce a second factor beside the traditional password for the user to access his account. While this method eliminates the need for user vigilance to protect him from phishing attacks, it introduces complexities in cost, usability and deployability features of the system as we will explain later.

2.3 Anatomy of phishing attacks and solutions: an overview

It is important to explain the phishing attack life cycle in order to categorize and explain the different phishing detection and prevention techniques that are found in the literature to mitigate or prevent phishing attacks. Figure 3 depicts a simple flow chart describing the life cycle of a phishing attack from the perspective of phishing detection techniques. Usually phishing attacks starts by sending messages (mostly phishing emails) to possible victims either randomly or to specific users (Spear Phishing), the first defense line in combating phishing attacks is to detect or identify the possible phishing message. The

detection process is a complex, broad and could incorporate techniques provided by the

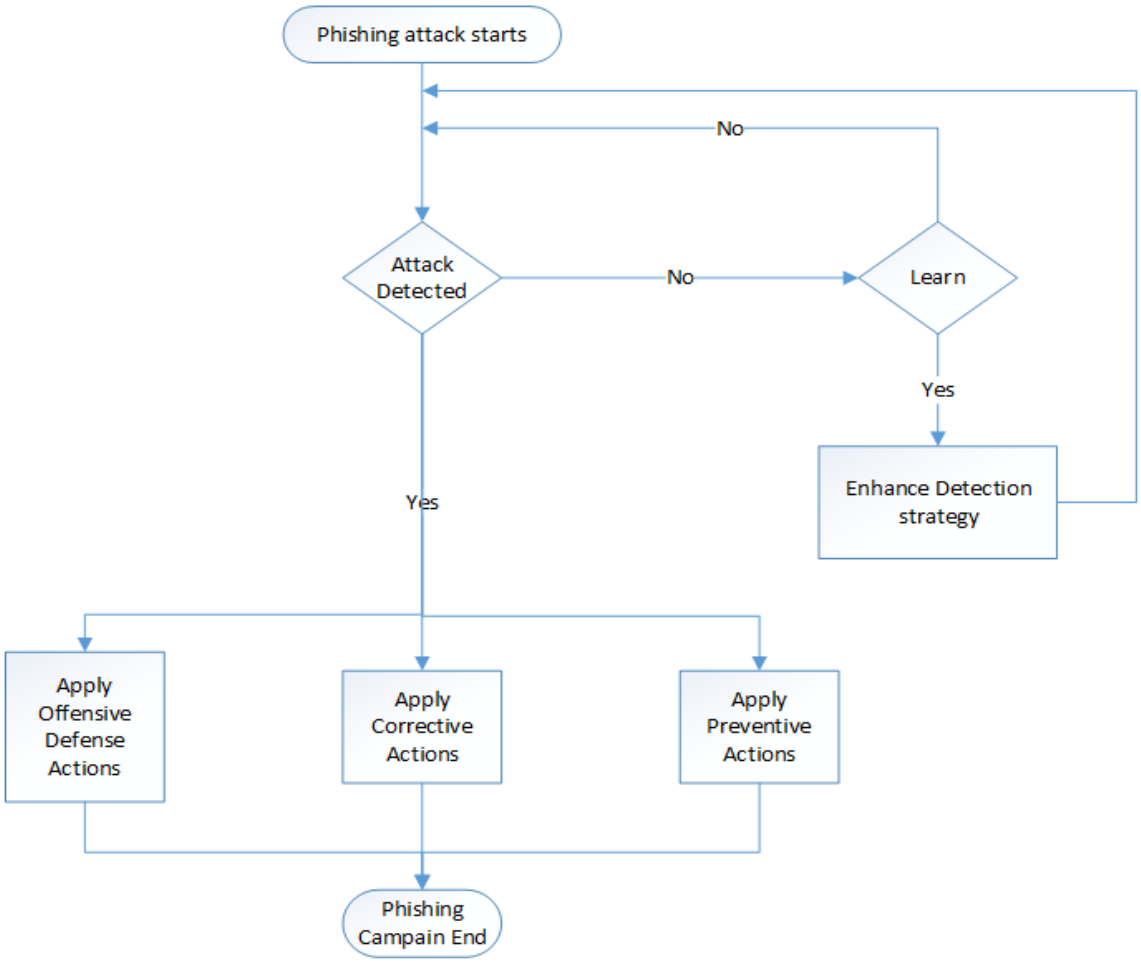

Figure 3: Phishing attacks flow chart (Detection Mode)

Service providers, end user client detection (classification and clustering) software in addition to user awareness and knowledge that depends mostly on user education and training about the phishing problem.

Once a phishing attack is detected, it is possible to take the following actions:

- **Offensive defense**: In this approach, it is possible to carry out a counterattack aiming at disrupting the phishing attack to make it less effective and to decrease the opportunity of the attackers to make use of the stolen credentials. One form of

such disruption is to flood the phishing web site with misleading credentials in an attempt to delay compromising the real stolen credentials.

- **Correction**: In this approach, the target is to stop the phishing campaign by deactivating the weaknesses (bugs and vulnerabilities) on different levels, such as those on the target web site or the email client. Corrections will focus on stopping the phishing attack from compromising more users, for example by suspending the hosting account of the website or removing phishing files.
- **Prevention**: the phishing prevention approaches take completely different strategies to protect user private data. In summary, they aim to protect the user account in case he falls as a victim to a phishing attack by introducing a second authentication factor in addition to the traditional user name/ password pairs. Figure 4 depicts a high-level overview of a phishing prevention strategy.
- **Learning:** this process aims at enhancing the accuracy and efficiency of the phishing detection tools. The learning process includes reporting phishing keywords and URLs to blacklisting services, adding content to classifiers to enhance their detection accuracy, filtering web sites as legitimate in local knowledge bases of a detection tool and contributing in enriching the features used to classify content in some approaches.

The mission of combating phishing attacks has two different directions; detection approaches and prevention approaches; at a simple level, the difference between detection and prevention techniques is that detection techniques are designed to inform the user that a phishing attack is detected and that his credentials might be compromised if he follows the phisher's directions. While prevention techniques actually attempt to prevent a

phisher from accessing the website on behalf of the user in case the phisher succeeded to compromise the user's credentials.

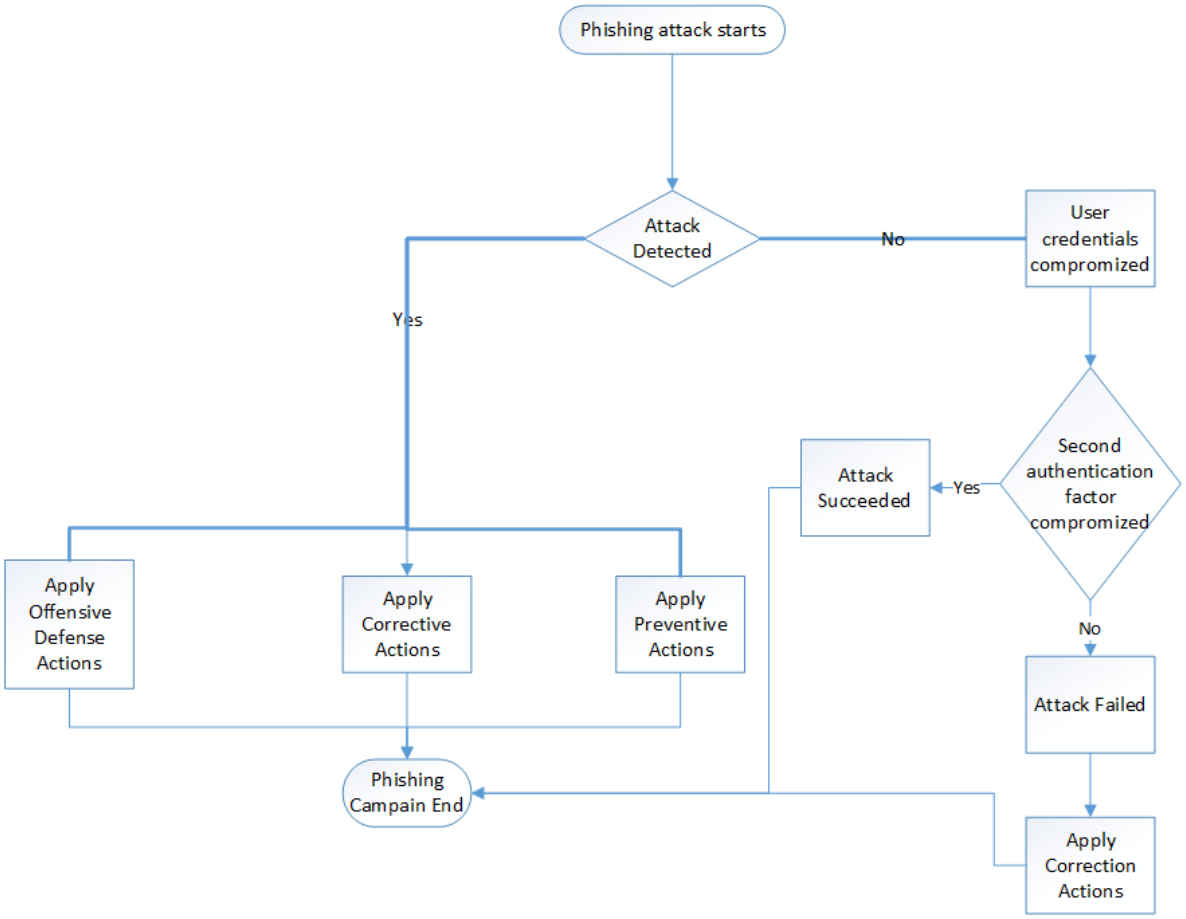

Figure 4: Phishing attacks flow chart (Prevention Mode)

2.4 Phishing detection

In the literature of phishing attacks and solutions, any attempt to identify or classify phishing attacks is considered a phishing detection technique, including:

- **User training approaches**: aim at educating users about phishing attacks and how to detect such phishing emails. Further discussion about the human factor in fighting phishing attacks is presented in section 2.6.

- **Software detectors and classifiers**: aim at automating the task of correctly classifying and may be blocking phishing content on behalf of the user himself, and thus mitigate the risks of relying on the user vigilance and knowledge in combating phishing emails or websites. Further discussion about software classifiers is presented in sections 2.8, 2.9 and 2.10.

Figure 5 depicts an overview of the phishing detection strategies as we reviewed in the phishing literature, detailed explanation and comparisons of those strategies is presented in the following sub sections.

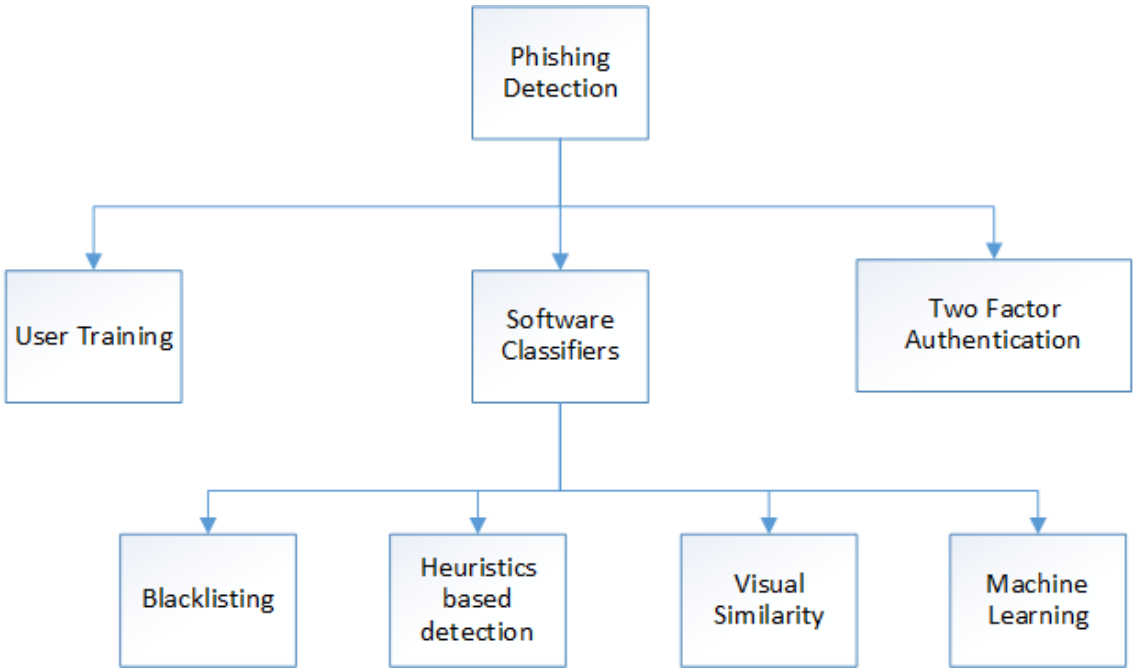

Figure 5: Phishing detection approaches

2.5 Evaluation Metrics of software classifiers

Since the following subsections explore a set of phishing detection techniques that were explored in the literature, those techniques are evaluated using a set of evaluation metrics that need to be explored here.

The task of detecting phishing attacks is usually treated as a classification problem, where the detection model goal is to detect phishing instances (email message or a web site) from a data set consisting of legitimate and phishing instances. The process of building the detection model usually incorporates a set of phases including data preprocessing, training and testing the model. The evaluation of the model effectiveness and efficiency in detecting phishing content is measured using a set of metrics which we briefly introduce here.

The confusion matrix of any binary classification problem (assuming a phishing classification task) is:

- $Np \rightarrow p$: is the number of phishing instances that are correctly classified as phishing.
- $Nl \rightarrow p$: is the number of legitimate instances that are incorrectly classified as phishing.
- $Np \rightarrow l$: is the number of phishing instances that are incorrectly classified as legitimate.
- $Nl \rightarrow l$: is the number of legitimate instances that are correctly classified as legitimate.

The most common evaluation metrics of phishing detection techniques are:

- True Positive (TP) rate: the percentage of correctly classified phishing content in relation to all phishing content in the data set. Formally, TP rate is defined as $TP = (Np \rightarrow p)/(Np \rightarrow p + Np \rightarrow l)$
- False Positive (FP) rate: the percentage of legitimate content that are miss classified as phishing content, given by : $FP = (Nl \rightarrow p)/(Nl \rightarrow l + Nl \rightarrow p)$

- True Negative (TN) rate: the percentage of correctly classified legitimate (not phishing) content in relation to all legitimate content in the data set, given by: $TN = (Nl \rightarrow l)/(Nl \rightarrow l + Nl \rightarrow p)$
- False Negative (FN) rate: the percentage of phishing content that are incorrectly classified as legitimate in relation to all existing phishing content in the data set, given by : $FN = (Np \rightarrow l)/(Np \rightarrow p + Np \rightarrow l)$
- Precision (P) — measures the rate of correctly detected phishing attacks in relation to all instances that were detected as phishing, given by: $P = (Np \rightarrow p)/(Nl \rightarrow p + Np \rightarrow P)$
- Recall (R) — equivalent to TP, given by: $R = TP = (Np \rightarrow p)/(Np \rightarrow p + Np \rightarrow l)$
- f1 score — Is the harmonic mean between P and R, given by: $f1 = 2PR/(P + R)$
- Accuracy (ACC) — measures the overall rate of correctly detected phishing and legitimate instances in relation to all instances, given by:

  $ACC = (Nl \rightarrow l + Np \rightarrow p)/(Nl \rightarrow l + Nl \rightarrow p + Np \rightarrow l + Np \rightarrow P)$

2.6 The human factor in phishing detection

This section presents and discusses the related work contributing in the education of users regarding combating the phishing attacks.

2.6.1 User behavior regarding phishing attacks

A set of research efforts have been conducted to explore the characteristics and behavior behind falling as a phishing victim, taking into account the users technical knowledge about security in general and about phishing in specific. In addition to users' behavior about phishing warnings presented by software classifiers.

(J. S. DOWNS et al., 2007) conducted a study on more than 200 users to specify the potential reasons that could make them fall as victims to phishing attacks, the results state that users who have knowledge about the definition of phishing were less likely to fall as victims. The study concluded that educating users about the phishing problem is more important than just warning them about the dangers and consequences of phishing attacks.

The study in (H. HUANG et al., 2009) concluded that users' ignorance of passive warnings about possible phishing attacks is one of the main reasons that lead them to fall as victims to those attacks. The study in (S. SHENG et al., 2010) introduced other factors that increase the probability of a user to be victim of phishing attacks, including gender and age of the user; for example, females are more likely to click on email links than males, and users aging between 18 and 25 years are more likely to fall victims than others. The authors justify their findings by the lack of knowledge among those categories.

2.6.2 Users interaction model of phishing attacks

The phishing content usually contributes to the user's decision in classifying that content as legitimate or phishing one; usually the phishing content (e.g. email messages) is divided into two parts, as depicted in figure 6:

- **Metadata**: are descriptive tags about the phishing content, include URL address of a website or the email address of the email sender and receiver. In general, it is a hard task for users to decide whether the meta data indicates a phishing content or not, for example, it is difficult to decide whether an IP address in the web site URL points to phishing web site or not. So users are not expected to analyze the Meta data as part of their effort to classify phishing attacks (W. D. YU et al., 2008); in fact, it is the task of automated software tools and classifiers to do so.

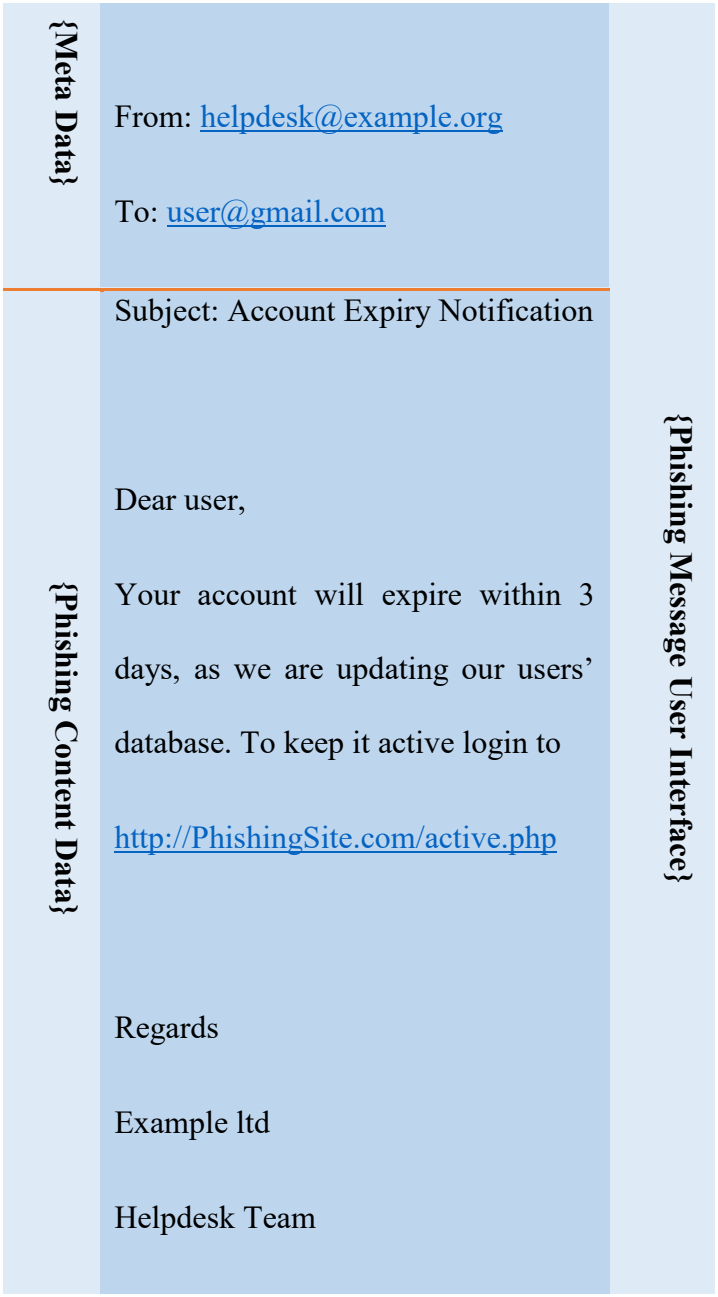

Figure 6: Sample phishing message content

- **Content data**: which is the real content of the phishing web site or email message. Modern phishing content applies social engineering and professional design skills to produce an exact copy of the legitimate websites and a convincing email messages. While users' awareness and knowledge will contribute in detecting such phishing content, we believe that the perfect solution is to design websites in such

a way that prevents phishing attacks regardless the user decision, as explained later in chapters 4 and 5.

2.6.3 Educating users about service policies

Different phishing detection solutions have been proposed and implemented to notify or warn the user about possible phishing attacks, including email classifiers and browser plugins. The bottleneck of those solutions is that they are dependent on the user behavior about those notifications and warnings, if the user simply ignores those warnings and follow the instructions in the phishing email and/or enters his credentials into the phishing website, then the user will certainly fall as a victim and the detection tools are simply useless.

Service providers are expected to educate their clients about phishing and social engineering attacks to enhance their capability to respond positively to phishing detection tools, a common solution is to educate clients through sending emails, SMS, brochures that clarify different aspects of possible phishing attacks. Those communication methods have been studied and evaluated in the literature of user awareness and education; (P. KUMARAGURU et al., 2007) concluded that periodic security notices are ineffective and usually fail to change users' behavior.

Service providers are also expected to apply strict IT policy regarding security in general, including plans that specify exactly how to respond in case phishing attacks succeed to compromise some of their accounts, or in case their security models are threatened. One solution is to shut down the compromised services, (T. MOORE AND R. CLAYTON, 2007) concluded that service takedown is a common approach in handling security problems.

Another direction in educating users is making them aware of the environment they work with by educating them thoroughly about email clients, web browsers, and mainly the technologies and services that they interact with or are rendered to users. For example, knowledge about SSL certificates and their validity, browser plugins, browser security warnings and email digital signatures will add a significant value for users to enable them to correctly decide whether a message content (along with the phishing detection tool warnings) is possibly a phishing one or not. While it is not logical that novice users are expected to learn all such technological hints, knowing about part of them could lead to a better user behavior and decision as they are an important part of the external information that contribute to the user interaction model with phishing content.

2.7 Phishing detection warning messages and their effectiveness

Phishing detection tools -such as email classifiers and web browser plugins- work on the principle of notifying the user about a possible phishing attack through displaying warning messages on his email client (though blocking or moving the infected email to junk folder) or by visual or textual warnings on the web browser.

There are two methods of warning the user about phishing attacks:

1. **Passive warnings**: where the warning message does not block the content area, i.e. the user is visually informed with clear warning that he is visiting or accessing a phishing content, as depicted in figure 7.
2. **Active warnings**: where the content data is blocked and the user is prevented from viewing or accessing it, as depicted in figure 8.

Figure 7: Internet Explorer's passive warning page against a suspected site. (http://blog.codinghorror.com/phishing-the-forever-hack)

Research studies about the effectiveness of both passive and active phishing warning messages preferred active warnings than passive ones. (S. EGELMAN et al., 2008) concluded that passive warnings are ineffective as 13% of the participants in their study paid attention to passive browser warnings, while active warnings increased the percentage into 79%. Another study (M. WU et al., 2006) stated that users ignore toolbar's passive warnings about possible attacks or unsafe content.

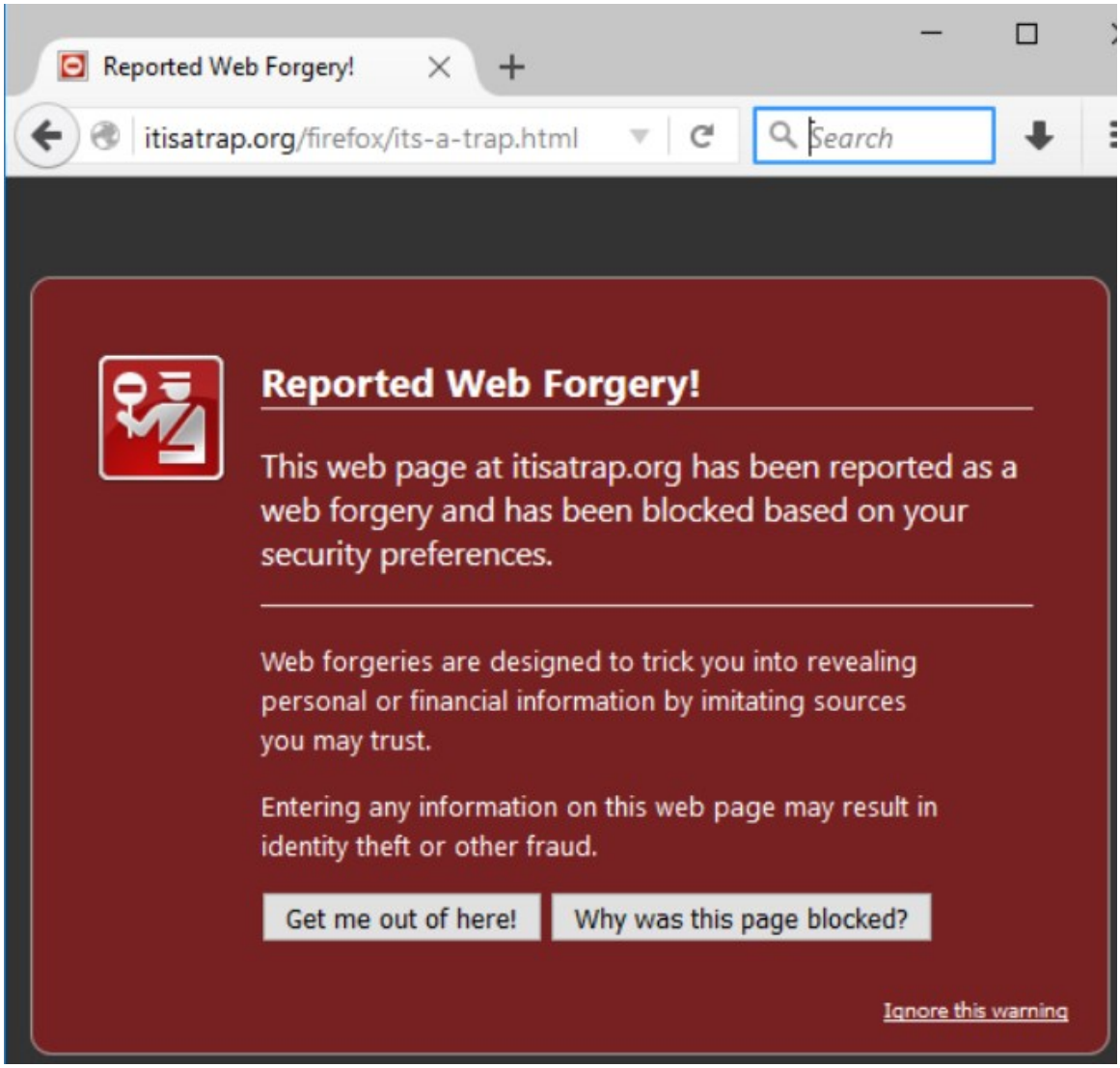

Figure 8: A sample active warning from Firefox browser

2.8 Black-list based phishing detection

A common approach in detecting phishing attacks is the black-list based detection techniques; where a list of known phishing content (URLs, keywords, IP addresses) is maintained and regularly updated with new identified phishing content. This approach works on the principle that any content that exists in the black list is directly considered a phishing content, and so the user is warned or the content is blocked directly. The black list is usually built from frequent reporting of a content that it is a phishing or suspicious content.

The main shortcoming of blacklisting is the time it takes for a phishing site to be blacklisted; the phishing site needs to be reported and then accurately identified (by the blacklist providers, e.g. phishtank.com) as malicious in order to avoid false positives. This shortcoming of blacklisting renders the technique weakness as phishing sites are now typically hosted for only about 12 hours – their uptimes have decreased over the years as indicated by APWG reports. The phisher's strategy these days is to host the phishing site for a few hours, attack as many users as possible during this period, and then move on to another phishing site. The blacklisting process simply cannot operate fast enough to be an effective defense for zero-hour phishing attacks, as novice users are attacked and maybe trapped before the blacklist becomes active. (S. SHENG et al., 2009) conducted an empirical analysis about phishing attacks and concluded that blacklists are ineffective against zero-hour phishing attacks, and were able to detect only 20% of them.

To take advantage of phishing blacklists, browsers need to get connected online with the blacklist service providers, submit the content and decide –upon the feedback from the blacklist provider- whether it is suspicious content and whether to warn the user passively or actively about this suspicion. In this context, we will introduce a set of well-known blacklists providers.

2.8.1 Safe browsing API by Google

Google defines its Safe browsing API as "a Google service that enables applications to check URLs against Google's constantly updated lists of suspected phishing, malware, and unwanted software pages." (GOOGLE SAFE BROWSING API)

Phishing detection techniques can make use of Google safe browsing API in two different modes:

1. **Safe Browsing API v3**: enables applications to download an encrypted table for local, client-side lookups of URLs, The Safe Browsing API is used by several browsers, including Google Chrome and Mozilla Firefox. Using this API gives the following advantages:
    - Privacy: API users exchange data with the server using hashed URLs, so the server never knows the actual URLs queried by the clients.
    - Response time: API users maintain a local cache of the hashed URLs in Google's suspected phishing, malware, and unwanted software lists; they do not need to query the server every time they want to check a URL.

    The major drawback of the Safe Browsing API v3 is its implementation complexity, according to (GOOGLE SAFE BROWSING API), Safe Browsing API v3 users need to:
    - Be aware of the internal structures of how the server stores hashed URLs in the phishing, malware and unwanted software lists, and implement the hashing and suffix/prefix expressions.
    - Periodically update their local cache of the hashed URLs. If there are updates, they also need to download the new lists of hashed URLs.
    - Download and compare the full hash value of URLs that are hit in the local cache.
2. **Safe Browsing Lookup API:** is an experimental API that enables applications to send URLs to Google's Safe Browsing service and check their status (e.g. phishing, malware, unwanted software). This API is easy and simple to implement; API users send a HTTP GET or POST request with the URLs, and the server responds with the state of the URLs.

Safe Browsing Lookup API drawbacks are:

- Privacy: URLs are not hashed, so the server knows which URLs API users look up.
- Response time: Every lookup request is processed by the Safe Browsing server. Google does not provide guarantees on lookup response time.

2.8.2 PhishTank

PhishTank is a free community site where anyone can submit, verify, track and share phishing data, once the submitted data is validated by the PhishTank as a phishing content, it is listed on the phishing blacklist and made available for public through browsing or using the free API. Figure 9 depicts a snapshot of the most recent phishing websites submitted to PhishTank.

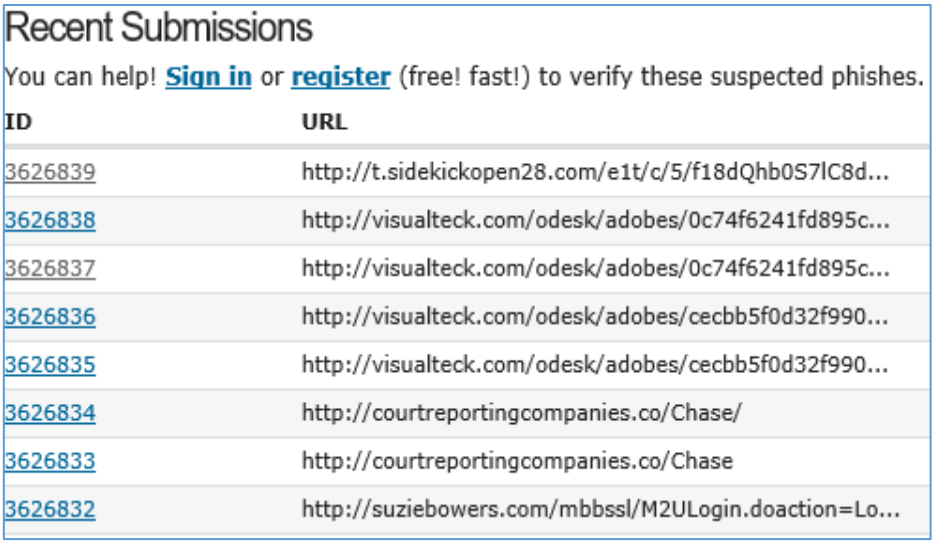

Figure 9: Sample phishing websites from PhishTank.com

2.8.3 PhishNet: Predictive Blacklisting

Phishing detection tools use blacklists to decide whether a given URL is listed as a phishing or not. This process is done by comparing the URL with the existing list of phishing URLs stored in the blacklist database provided by the blacklist providers, which implies that any changes in a suspicious URL may result in failing to match it

with existing blacklisted URLs. PhishNet (P. PRAKASH et al., 2010) tries to solve the exact match problem found in blacklists. PhishNet produces the possible variations of any blacklisted URL using a set of heuristics applied to the parent URL, some of them are:

Heuristics1, Replacing TLDs (Top Level Domain): for each new URL that enters a given blacklist, replace the effective TLD of the URL with 3,209 other effective TLDs that form the candidate child URLs that need to be validated. For example, the URL http://www.phishingurl.com will be replaced by http://www.phishingurl.com , http://www.phishingurl.in, http://www.phishingurl.org …etc. and all those variations are validated against the black list.

Heuristics2, IP address equivalence: Phishing campaigns from the same source IP address may share the directory or path structure among each other, so blacklisted URLs originating from the same IP address are clustered together and new possible phishing URLs are created by considering all combinations of hostnames and pathnames.

Heuristics3, Directory structure similarity: create multiple permutations of URLs that have similar directory structure. For example: http://www.site1.ps/online/ebay.html and http://www.site2.ps/online.paypal.com would result into creating the following children URLs: http://www.site2.ps/online/ebay.html and http://www.site1.ps/online.paypal.com.

2.9 Heuristics Based Phishing Detection Techniques

Wikipedia defines the heuristic as "*any approach to problem solving, learning, or discovery that employs a practical method not guaranteed to be optimal or perfect, but sufficient for the immediate goals*", regarding to phishing attacks we can define heuristics

as any characteristics that usually exist in phishing content but are not guaranteed to always exist.

The idea behind heuristics based phishing detection techniques is to develop tests based on the identified phishing heuristics. While those tests could be effective for detecting zero-hour attacks that blacklisting usually fails to detect, they incorporate the risk of misclassifying legitimate content.

Most web browsers and email systems are supported with phishing protection mechanisms, such as heuristic tests that aim at detecting phishing attacks, including Mozilla Firefox, Internet Explorer and MS outlook. In this context, we introduce some of the heuristic based phishing detection techniques.

2.9.1 Spoofguard

Spoofguard (SPOOFGUARD, 2005) is a web browser plugin that detects HTTP and HTTPS phishing sites, works as a toolbar -developed by Stanford University- that analyzes the HTML content of the suspected site and alerts the user of a possible suspicious attack. The toolbar extracts certain heuristic anomalies in the parsed HTML content of the suspected website and weights them against a threshold value; examples of test heuristics that Spoofguard applies to HTML content are:

1. Existence of URLs that are similar to known (white-listed) website URLs. for example, if the content contains the URL www.yaho0.com which is similar to the popular www.yahoo.com URL, Spoofguard will consider it as a phishing attribute.
2. Existence of cloaked URLs: A cloaked URL (RFC2396) is similar to regular URL forwarding, except that your URL will never change in the address bar. When you use cloaked URL forwarding, your domainname.com forwards to

your actual URL, but yourdomain.com stays in the address bar. An example of a cloaked URL is http://www.yahoo.com@www.phishingurl.com. If Spoofguard detects any cloaked URL in the phishing content, it will consider it as a phishing attribute.

3. Any difference between the URL anchor text and the URL itself is considered a phishing attribute by Spoofguard. For example,

   www.yahoo.com.

2.9.2 PhishGuard

PhishGuard (P. LIKARISH et al., 2008) is a web browser plugin that builds its detection model on the assumption that phishing web sites will not verify the user names and passwords submitted by victims, and just stores them for the attacker benefit without any validation. Based on this assumption, the authors in (P. LIKARISH et al., 2008) implemented a proof of concept using HTTP Digest Authentication.

In order to identify a suspected web page, PhishGuard follows the following steps:

1. Once the user opens a web page, and the page sends an authentication request (based on Digest authentication), and if the user submits the authentication form, then PhishGuard starts its testing procedures.
2. PhishGuard plugin will send the same user id with random password for n times to the server of the web page.
3. If the web page returns HTTP 200 OK message, then the web page is a phishing and it just accepts any user id/ password combinations.
4. If the web page returns HTTP 401 Unauthorized message, then two scenarios are possible:
   - The web page corresponds to a legitimate site.

- The web page is a phishing site that always returns failure authentication response.

5. PhishGuard sends the real credentials (originally entered by the user) to distinguish between the two possibilities in step 4 above.
6. If the web page responded with a HTTP 200 OK message for the real credentials, then it is a legitimate web page.
7. If the web page responded with a HTTP 401 Unauthorized message, then two scenarios are possible:
    - The user submitted the wrong username/password credentials
    - The web page is a phishing page.
8. To verify that the submitted password is not wrongly typed by the user, PhishGuard stores password hashes in a file and verifies future login requests against it:
    - If the hash of the entered password matches any entry in the file, then PhishGuard concludes that the password was correct, and the site was a phishing website.
    - If no match was found, then PhishGuard would conclude the supplied password is wrong, and the user is then alerted to correct his/her password.

By simply analyzing the PhishGuard phishing detection steps described above, it is easy to conclude its weak effectiveness against phishing attacks for the following reasons:

1. PhishGuard starts its steps after submitting the user credentials, this means that if the web page is a phishing page then the user has actually fallen as a victim and the phishing site received his credentials.

2. PhishGuard could be useful for notifying users if they again visit the phishing site, which is usually not practical as most phishing attacks have a limited life period and usually finish within 24 hours.
3. The PhishGuard strategy could be totally broken if the phishing web page employs the Man in the Middle Attack strategy; because it will be capable of returning the right authentication response from the original web site, (recall that MITM attacks generate real requests to the original website on behalf of the user). In this case, the web page will be certainly classified as legitimate by the PhishGuard plugin, making it practically not only useless, but harmful to users.

2.9.3 Phishwish

Phishwish (D. L. COOK et al., 2008) is a stateless phishing filter for email messages that applies an 11 heuristic rules to identify or classify suspected phishing emails, the proposed rules are (positive results means the email is suspected to be phishing):

1. If any URL that appears in the email is a login page but not identified as a business login page – this check is performed through the data extracted from search engines, then the result is positive.
2. If the email is HTML-formatted and any URL title uses TLS or HTTPS and the HREF attribute of the URL does not include TLS or HTTPS, then the result is positive.
3. If any included URL has an IP address as host name, then the result is positive.
4. Existence of an organization name in the URL path only and not existence of that organization name in domain part of the URL will lead to a positive result.
5. If the domain in the URL title is not found in the domain of the corresponding HREF attribute, then the result is positive.

6. If the emails SMTP header does not include the organization's name, then the result is positive.
7. If inconsistencies are found in a non-image URL's domain portion, the result is positive.
8. If inconsistencies are found in Whois records of non-image URL's domain portion, the result is positive.
9. If inconsistencies are found in image URL's domain portion, the result is positive.
10. If inconsistencies are found in Whois records of image URL's domain portion, the result is positive.
11. If any URL in the email is inaccessible, then the result is positive.

Phishwish decides whether the email message is a suspected phishing email by evaluating the above 11 rules against the email message content and headers; each applicable rule is given a weight and a total score is calculated. If the total score exceeds a threshold (typically 50%) then the email message is considered a suspected phishing email.

Phishwish follows the way of Apache spam filtering technique SpamAssassin (J. MASON, 2005), with far less rules than those of SpamAssassin (975 rules), in addition, it offers better protection against zero-hour attacks than blacklists.

2.9.4 CANTINA

CANTINA (Y. ZHANG et al., 2007) is a toolbar for Microsoft Internet Explorer that uses a content based approach to detect phishing websites, Cantina's model employs Term Frequency-Inverse Document Frequency (TF-IDF), search engines and a set of heuristic rules to classify suspicious phishing pages and to reduce false positives.

TF-IDF is a popular metric in the fields of text mining and information retrieval, it evaluates the importance of a term (or a word) in a document that resides in a collection

or corpus. The term's TF-IDF value increases proportionally to the number of times it appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general like stop words or common language words.

The TF-IDF value of a given term *i* in document *j* is the product of TF and IDF values of the same term *i.* TF is calculated by the following equation:$TFij = \frac{Nij}{\sum_{m=1}^{k} Nm,j}$

Where Ni,j denotes the frequency of the term i in document j, and k is the number of terms in document j.

IDF is given by the following equation: $IDFi = \log\left(\frac{|D|}{|Di|}\right)$, Where |D| is the cardinality of all documents available in a given corpus, and |Di| is the cardinality of all documents containing the term *i* in the same corpus. So, TF-IDF values for each term *i* is calculated by: $TF - IDFi, j = TFi, j \, . IDFi$

CANTINA model applies the following actions to detect phishing sites:

1. Calculate the TF-IDF value for each term in the web page.
2. The model chooses the top 5 terms with highest TF-IDF values in the web page.
3. The chosen terms in from step 2 are submitted to a search engine, and the domain names of the top five results are stored.
4. If the web page domain name is one of the domain names stored in step 3, then it is a legitimate web page, otherwise it is a suspicious or phishing web page.

To reduce false positives, CANTINA applies the following heuristic rules:

1. If the age of the web page domain name is greater than a year, then it is likely to be a legitimate web page.

2. If the URL of the web page (or any link in the web page) contains – or @ characters, then it is a phishing web page.
3. If the URL of the web page (or any link in the web page) contains more than five dots, then it is a phishing page.
4. If the web page contains HTML forms, then it is likely to be a phishing page.
5. TF-IDF evaluation of the web page.

Each heuristic is given an empirical weight value, and the state (S) of each page is calculated by:$S = f(\sum \quad Wi.Hi)$, where f is threshold function that returns 1 or -1 for legitimate or phishing result, Wi is the weight of the heuristic and Hi is the heuristic value.

While CANTINA works well for multi-lingual content as the set of heuristics are not language dependent, in our point of view, CANTINA could face the following problems that will affect its effectiveness in detecting phishing attacks:

1. System performance issue: the incorporation of calculating the top 5 TF-IDF words and then searching for them using a search engine will degrade the response time of a browser that decides to use such a toolbar.
2. Modern phishing pages may employ text on images, which makes the algorithm of CANTINA more difficult if not non-applicable.

2.10 Applying Visual Similarity Techniques in Phishing Detection

The phishing detection techniques analyzed so far process the phishing content (email header and body, web page source code) in order to infer or extract specific patterns, properties or behavior on a suspected phishing email or web page to be able to decide whether to classify it as a phishing content. In this section. We will review and analyze a set of phishing detection techniques that depend on the visual similarity principles, i.e.

based on the visual appearance of a web page, rather than analyzing the source code or network level information of a phishing message.

Afroz et al. proposed PhishZoo (A. AFROZ & R. GREENSTADT, 2011) to detect phishing attacks based on content similarity between legitimate sites and malicious sites and the assumption that malicious sites tends to mimic a legitimate site appearance to persuade users that they are accessing the expected sites. The basic principle of PhishZoo is to maintain profiles of possible legitimate sites that the user selects and store those profiles to compare malicious sites against them. The approach of PhishZoo is illustrated in figure 10, once a website is loaded, it is matched with the stored profiles, and if a match is found (the URL and SSL of the website match those of a stored profile) then it is a legitimate site. Otherwise, the loaded site contents will be matched against PhishZoo's profiles.

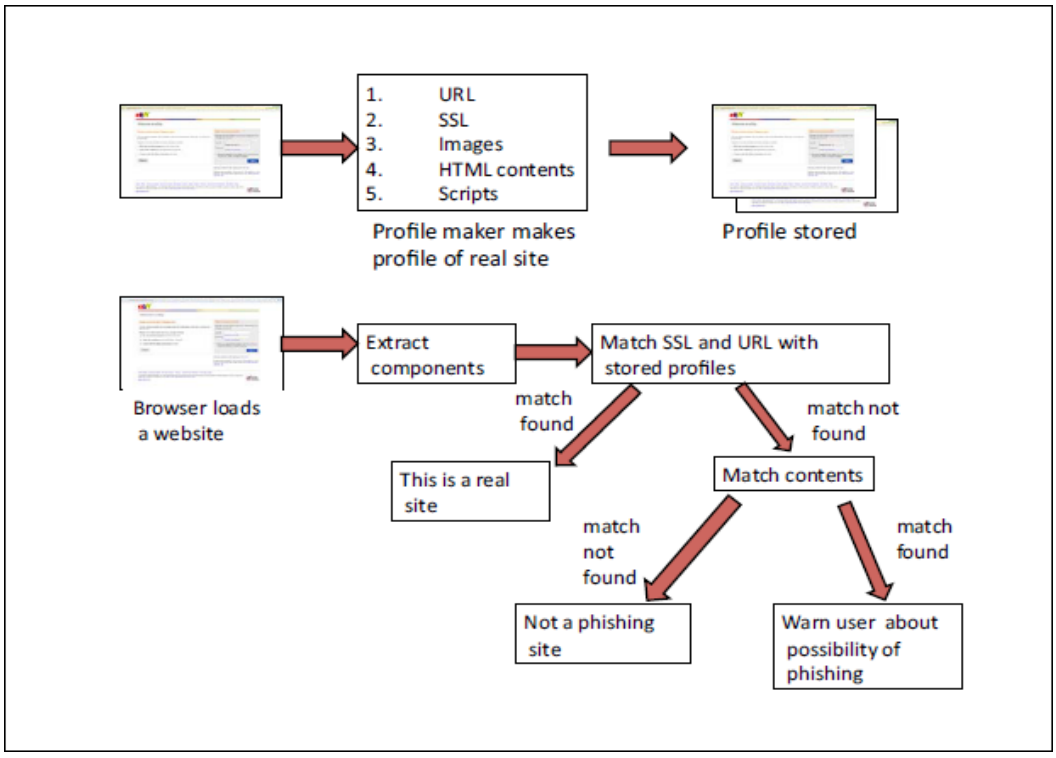

Figure 10: PhishZoo detection approach. (A. AFROZ & R. GREENSTADT, 2011)

The matching process in PhishZoo utilizes website contents including (images and HTML elements) and applies Scaling Invariant Feature Transformed (SIFT) algorithm with fuzzy hashing to identify phishing websites. While this approach can detect zero hour phishing attacks, attackers can easily hinder its effectiveness by changing the content structure of their phishing site.

Raw et al. (R.S. RAO, AND S.T. ALI, 2015) proposed a phishing detection approach based on a combination of Whitelisting and visual similarity based techniques. The web page is matched against a white list of legitimate websites, and if the result is negative then the approach uses SURF (Speed Up Robust Features) (H. BAY et al., 2006) detector to extract discriminative key point features from both suspicious and targeted websites. Then they are used for computing similarity degree between the legitimate and suspicious pages as depicted in figure 11.

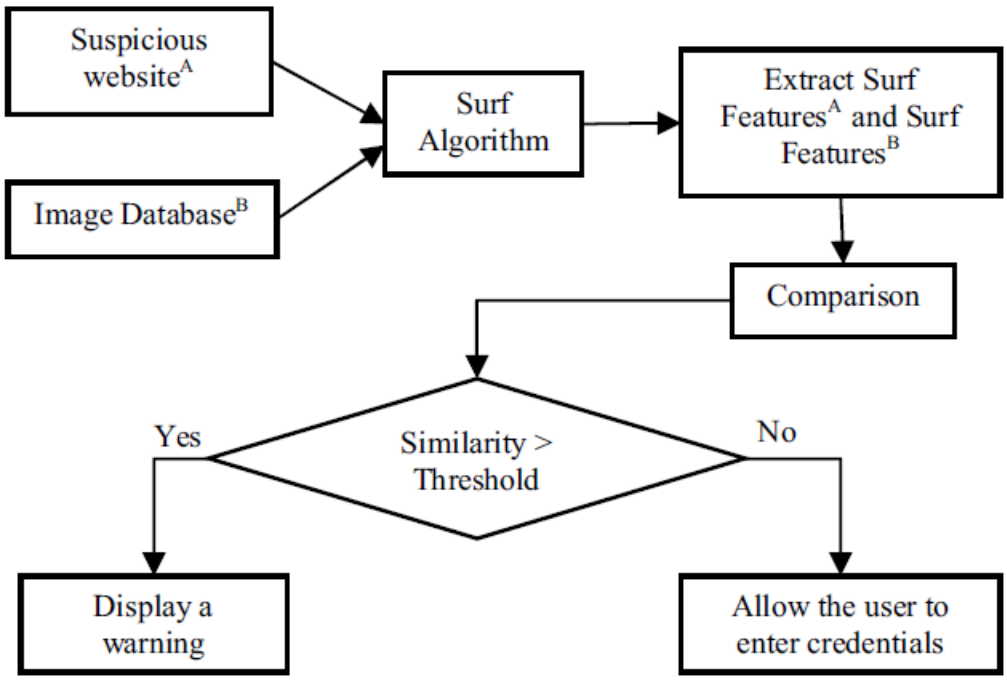

Figure 11: SURF-based phishing detection approach (R.S. RAO, AND S.T. ALI, 2015)

While this approach is effective in case the malicious site is very similar to a target site, its weakness –like most visual similarity based approaches- is in the delay imposed by the matching process.

Other approaches that are based on visual similarity focused on matching the DOM (Document Object Model) of the suspected web page with that of a Whitelisted one. DOMAntiPhish (H. BAY et al., 2006) is a browser plugin that compares the Document Object Model of both legitimate website and suspicious website to detect phishing; it tries to find the longest similar sub tree within the tag structure of both web pages, and uses a threshold to decide whether it is a phishing site or not.

2.11 Detecting Phishing Attacks using Data Mining

Data mining is the process of extracting information or discovering hidden patterns from data, using the techniques of artificial intelligence, machine learning, statistics, and database systems concepts. Data mining is part of the Knowledge discovery (KD) process for extracting or discovering patterns from data, the extracted patterns should be novel, valid, useful and understandable (USAMA FAYYAD et al., 1996). The KD process is carried out using a set of iterative steps as depicted in figure 12.

The steps are initiated by understanding the problem and the data, this step focuses on the understanding of objectives and requirements from a business perspective, learning the relevant prior knowledge about the problem domain and the goals of the end user of the discovered knowledge. The second step is a data pre-processing phase to prepare the prior knowledge data for the data-mining step through which the target knowledge will be discovered, evaluated and then presented as a useful and easy to use information.

Many proposed phishing detection approaches consider the detection process as a document classification or clustering problem; where a variety of algorithms and

techniques from the field of data mining are applied to a suspected content; including K-NN (K Nearest Neighbor), KMeans, Decision Trees, Random Forest, SVM (Support Vector Machine) and Density-Based Spatial Clustering of Applications with Noise (DBSCAN).

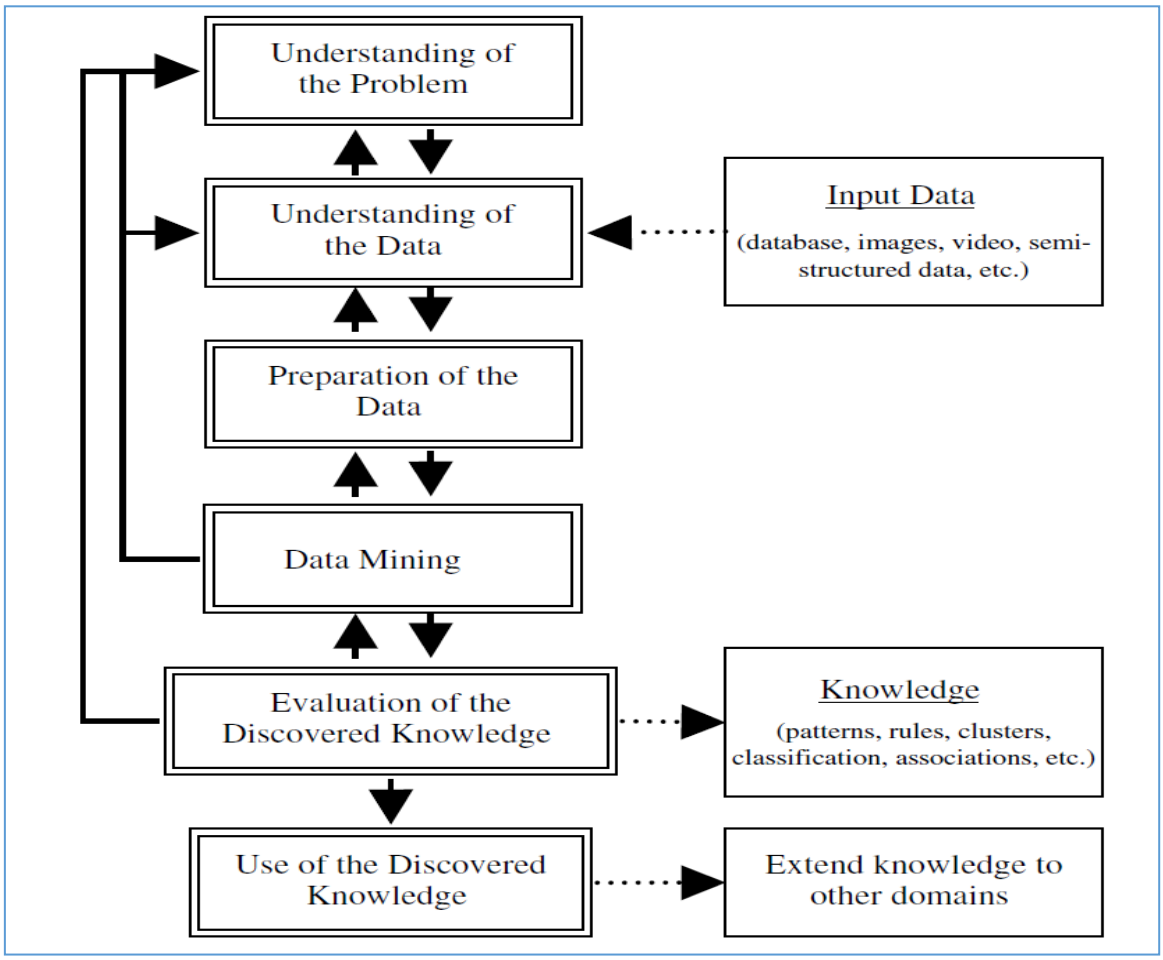

Figure 12: Knowledge Discovery Process (PAL et al., 2005)

In (SAMI SMADI et al., 2015), the authors proposed a model that utilizes 23 hybrid features of the email header and body extracted from about 10000 emails. The emails are divided equally between ham and spam emails, their model applied J48 classification algorithm to classify phishing and legitimate emails and concluded with an accuracy of 98.11% and false positive rate of 0.53%.

Another study (F. TOOLAN AND J. CARTHY, 2009) applied a two-phase classification model of emails. In the first phase, a set of classification algorithms (C5.0, Naive Bayes, SVM, Linear Regression and K-Nearest Neighbors) are used to classify legitimate and phishing emails, common evaluation metrics are used to evaluate each algorithm including accuracy, precision, recall and F-score. The algorithm with best classification results was C5.0 with an average accuracy rate of 97.15%, average precision of 98.56%, average recall of 95.64% and average F-score of 97.08%. in the second phase, the emails that were classified as legitimate in the first phase were input to an ensemble classifier.

The authors in (MAYANK PANDEY AND VADLAMANI RAVI, 2012) proposed an email classification model that exploits 23 keywords extracted from the email body, the proposed model was tested using a set of classification algorithms, including multilayer perceptron, decision trees, support vector machine, probabilistic neural net, genetic programming, and logistic regression. The best classification result was achieved using genetic programming with a classification accuracy of 98.12%.

The study (SUNIL B. RATHOD AND TAREEK M. PATTEWAR, 2015) applied the Bayesian classifier for phishing email detection, evaluated in terms of accuracy, error, time, precision and recall. The model resulted in accuracy of 96.46%.

The authors in (LEW MAY FORM et al., 2015) applied Support Vector Machine classifier to classify emails using a set of 9 structure-based and behavior-based features. The model achieved 97.25% accuracy in results, however its weakness is in its relatively small training dataset (1000 emails with 50% spam and 50% ham).

The authors in (TAREEK M. PATTEWAR AND SUNIL B. RATHOD, 2015) proposed an email classification algorithm by integrating Bayesian Classifier and phishing URLs

detection using Decision Tree C4.5, their approach achieved 95.54 % accuracy, which is better than the accuracy of 94.86% that was achieved using Bayesian classifier.

The study in (PRAJAKTA OZARKAR & DR. MANASI PATWARDHAN, 2013) used Random Forest and Partial Decision Tree algorithm for spam email classification, the authors applied a set of feature selection methods in the pre-processing step including Chi-square and Information gain, they achieved accuracy of 96.181% with Random Forest and 95.093% with Part.

The authors in (GAURAV KUMAR TAK AND GAURAV OJHA, 2013) proposed a browser knowledge-based compound approach for detecting phishing attacks, the proposed model analyses web URLs using parsing and utilizes a set of maintained knowledge bases which store the previously visited URLs and previously detected phishing URLs. The experimental results indicated 96.94% accuracy in detecting phishing URLs with a little compromise in degrading the browser speed.

The proposed phishing detection model described in (C. WHITTAKER et al., 2010) applies data mining techniques and end users' experience. The authors described an anti-phishing solution implemented by Google to rapidly and promptly classify pages while maintaining low false positives (below 0.1%), and the classified phishing pages are linked directly to the blacklist managed by Google Safe Browsing API (as described earlier).

The proposed methodology can be summarized as follows:

1. The classification task is initiated by Gmail users when they manually classify a suspected email as Junk.
2. Google considers the links in the junked emails as a phishing candidates only if the same links were classified as junk by many users to avoid users' abuse.

3. When a URL is classified as suspect by Google, a set of features are extracted from the URL, including:

- Whether the host name is an IP address.
- Number of sub-domains included within the URL (more sub-domains indicate it is a phishing URL).
- Whether the URL contain certain tokens; usually phishing URLs contains special token including trademarks of the attack target website.
- A set of features are extracted from suspected URL's page contents, including:
    - The existence of password fields.
    - Highest TF-IDF terms.
    - Existence of objects whom source is from different domains; as many phishing websites link to images from the target website.

4. The extracted features are fed to a machine learning classifier, and the output ranges is from 0.0 to 1.0 where 0.0 means a certain phishing website and 1.0 means a certain legitimate one.
5. The same algorithm is applied to webpages while being crawled by the google search engine crawlers.

2.12 Summary and conclusions

Phishing attacks have been thoroughly analyzed by academic researchers and the IT industry in the sake of developing anti-phishing solutions, those solutions range from user education strategies to developing advanced solutions that apply advanced data science techniques including data mining, visual similarity, machine learning and artificial intelligence. The first defense line against phishing attacks is to educate users and improve their security awareness, followed by applying advanced techniques to correctly detect phishing messages and applying the correct action to protect users.

Unfortunately, educating users and relying on phishing detection techniques are still not mature enough to stop the success of phishing attacks, and we think it is very important to redesign the web authentication schemes so that they play a major role in fighting phishing attacks, on the assumption that while phishing detection techniques play an important role as a first defense line against phishing attacks, it is the website vendor's responsibility to protect their user's data and accounts, and this could be achieved by developing robust authentication techniques that protects the user even when he falls a victim to phishing attacks.

3 Web authentication techniques

Passwords have been dominating all other methods of end user authentication over years, especially in the field of web authentication. While web technology evolves in an exponential trend in technology and usability, passwords reproduce themselves and are present in every new website that comes to existence.

Intensive research studies concluded that passwords are vulnerable to threats by security problems (R. MORRIS AND K. THOMPSON, 1979). (A. ADAMS AND M. SASSE, 1999) concluded that passwords are hated by users. Huge efforts have been conducted to propose schemes that replace traditional passwords or at least augment their security weaknesses; but unfortunately, no definitive solutions have been proposed to replace the passwords scheme at all; perhaps the syndrome of security, usability and deployability is the main reason why passwords still dominate the web authentication. Most proposed alternative authentication schemes incorporate authentication steps that compromise users' usability while provide more secure authentication.

However, security threats especially on web-based applications that aim at compromising users' private data and or their financial transactions impose a situation where all parties (vendors and users) of a web service to accept a less usable authentication scheme in order to maximize users' security. To achieve that, a set of authentication schemes have been proposed to protect users and vendors from security breaches and attacks especially phishing attacks.

In this chapter, we explore and categorize a set of techniques and proposals of authentication schemes that aim at enhancing users' security into a higher level, augmenting the traditional plagued-to-security-problems passwords. In this review, we explore and analyze a set of schemes that were proposed depending on the three authentication methods depicted in table 1.

Table 1: Authentication methods

Authentication method	Details
Something the user knows.	The traditional user name/ password scheme.
Something the user possesses.	like One-time password, smart card, hardware tokens.
Something the user is.	User's biometric identities like finger prints and iris.

The authentication methods that rely on the techniques in table 1 are a one point of failure solutions, as they may be compromised by phishing attacks, physical control, eavesdropping or brute force attacks. In addition, the biometric based authentication usually incorporates the usage of special hardware and software equipment that could be either unavailable or expensive for both the end user and the web service vendor.

3.1 Methods of enhancing password-based authentication

A set of work-around solutions are found in the literature to mitigate the weaknesses of password-based authentication techniques, here we review some of those solutions.

3.1.1 Encrypted Password Managers

Encrypted password managers are piece of software (usually a browser plugin) that offers to remember each password entered into a webpage, and could encrypt it with a master password that is known only to the user; after that, the password manager will fill in the username and password fields automatically each time the user revisits the web page. A set of password managers are already available; we explore some of them here for demonstration purposes:

- **Mozilla Firefox:** The Mozilla Firefox browser (MOZILLA FIREFOX) contains a built in feature that automatically offers to remember passwords entered into web pages, and optionally encrypts them with a master password that is known only to the user, and used later as an encryption key for the saved user passwords.

It also offers to synchronize passwords with other user machines by storing the managed passwords in the cloud and then being synced with all user other machines identified by the user's master password.

- **LastPass (**LASTPASS**):** is another password manager that follows the same principle of Mozilla Firefox of encrypting and remembering user passwords, it is a plugin that is interoperable with a variety of browsers and provides cloud storage and syncing of user encrypted passwords with a master password.

In general, encrypted password managers make it easier for the user to maintain his passwords without the need to remember them at all, all the needed from the user is to remember the master password if he wishes to access the website from a different machine. However, they suffer from the following weaknesses:

- Deployability weaknesses: some of them are browser specific such as the proprietary Mozilla Firefox browser.
- Losing the master password is catastrophic; all user accounts could be compromised in case the master password is compromised.
- They are still vulnerable to all security breaches that threaten the password-based authentication, as the master password is still a password!

3.1.2 Proxy Based Authentication Schemes

Proxy based authentication schemes place a proxy between the user's machine and the server, such that the proxy connects to the server on behalf of the user request. One reason for adopting such schemes is to enable user secure login despite their infected (malware) machines. In this context we review briefly two sample proxies: URRSA (D. FLORÊNCIO AND C. HERLEY, 2008) and Imposter (A. PASHALIDIS AND C. J. MITCHELL, 2004).

URRSA authenticates users to the end server using a pre-shared one-time codes carried on a sheet of paper. at registration the user enters the password "P_j" for each account "j" to be visited; which is then encrypted at the proxy with thirty different keys "K_i", giving $C_i = E_{Ki}$ (Pj): The C_i act as one-time codes which the user prints and carries, the codes are generally 8-10 characters long. The keys only stored at the proxy. At login the user visits the proxy, indicates which site is desired, and is asked for the next unused code. When he enters the code, it is decrypted and passed to the end login server. The proxy never authenticates the user, it merely decrypts with an agreed-upon key, the code delivered by the user.

The obvious weakness points of URRSA are in its scalability and usability features; an active user with tens of accounts will be exhausted with sheets that contain 30 keys for each account.

Impostor enables users to authenticate to their website from potentially infected machines by intermediating the traffic between the user's machine and the server. The user needs to register his accounts on Imposter proxy and agrees on a shared secret phrase for subsequent access to the proxy, also the user shares the websites along with his password for each one with the proxy server. When he needs to access his website, the user is required to change his browser proxy settings to be directed to the Imposter proxy and then login to the proxy by entering parts of the secret phrase.

This scheme eliminates the need for memorizing user passwords, and even makes it easier to challenge the user against the secret phrase by requesting him to fill parts of the phrase only. However, an attack on the Imposter proxy itself (dictionary attack) could result in compromising all user accounts

3.1.3 Federated Single Sign-On

This authentication scheme exploits the principle of the digital identity server that offers the service of identifying a user identity for the benefit of other relying parties. digital identity platforms allow users to log onto relying websites, applications, mobile devices and gaming systems with their existing identity, including OpenID (D. RECORDON AND D. REED, 2006) and Facebook Connect (FACEBOOK CONNECT, 2016) in addition to many others like Microsoft, Google and Yahoo.

The most popular federated sign-on protocol is OpenID, which allows any web server to act as Identity Provider to any server desiring authentication- known as Relying Party. When the user wishes to access a relying website, he will be authenticated using his OpenID digital identity, releasing him from the burden of memorizing and typing a set of passwords corresponding to every website he visits.

While OpenID reduces the security risk to the authentication between the user and his identity provider, it could turn into a complete compromise of all user's accounts in case this authentication has been compromised.

Facebook Connect is another popular federated single sign-on scheme that uses Facebook as an identity provider for users; such that users are redirected to Facebook to enter their Facebook account credentials (or click on a confirm button in case the user is already logged in to Facebook) to sign in to their relying websites. Under the hood Facebook Connect is based on OAuth (D. RECORDON AND D. HARDT, 2010), giving relying parties access to the users' profiles on Facebook, which is considered an added value for those relying parties to adopt such an authentication scheme, but at the same time making users' private data partially open to service providers.

3.2 Two-factor authentication (2FA)

The idea behind two-factor authentication (TWO FACTOR AUTHENTICATION) is to verify the user identity through more than one evidence, for example, the user is verified through a password and a hardware or software token. This scheme success point is in the assumption that it is far more difficult for an attacker who compromises the user password to compromise the second authentication factor. While this authentication scheme is popular and already put in production in many popular web services, many obstacles arise when analysing the key concepts of two factor authentication. In this context, we summarize some of those obstacles:

1. Cost considerations: implementing two-factor authentication usually introduces extra operational cost, for example, when using SMS or phone calls as a second factor.
2. Physical and logical security considerations: there should be extra efforts for both the end user and the web service provider in order to protect the second factor from being compromised, for example hardware tokens may be stolen; software tokens and their generator applications could be vulnerable to threats.
3. Availability considerations: the end user is required to keep the second authentication factor available with him each time he wishes to access the web site; this requirement could result in logistics problem for the user especially for hardware tokens.
4. Usability considerations: implementing two-factor authentication in a web site usually makes it less usable for users; for example, the user needs to do extra actions and steps to complete his authentication process, such as scanning a QR code, plugging a hardware token, installing a mobile application, generating and entering software code, etc...

5. Advanced phishing attacks: the study in (ALEXANDRA DMITRIENKO et al., 2014) concluded that 2FA schemes are vulnerable to organized and intelligent attacks that concentrate on the user registration step in 2FA schemes.

Despite the above usability and deployability shortcomings of 2FA schemes, giant companies like yahoo, Google and Facebook implement 2FA and strongly recommend their users to enable this authentication option on their accounts, as we will discuss in the next section.

3.2.1 Current two-factor authentication techniques

A set of authentication techniques that supplement the traditional password authentication are found in the literature, in this section we will review and analyse some of these techniques.

3.2.1.1 A Novel Anti Phishing framework based on Visual Cryptography

The authors in (DIVYA JAMES AND MINTU PHILIP, 2012) proposed an anti-phishing authentication technique based on Visual Cryptography. The proposed scheme technique is to generate an image captcha from the user's information that is collected in the registration phase, then the generated image is divided into two shares, one for the user and the other is maintained in the server. Then, the user's share and the original image are sent to the user for later verification during login phase.

The following steps take place when the user tries to login to the web application:

- The user enters his user name and his share.
- Once the user name and user share are submitted, the web server will stack the received user share and the share maintained in the server for that user to generate the image captcha.
- The generated image captcha is sent back to the user for verification.
- The end user then enters the text displayed in the image captcha.

The user registration process for this authentication technique is depicted in figure 13, and user logon process is depicted in figure 14.

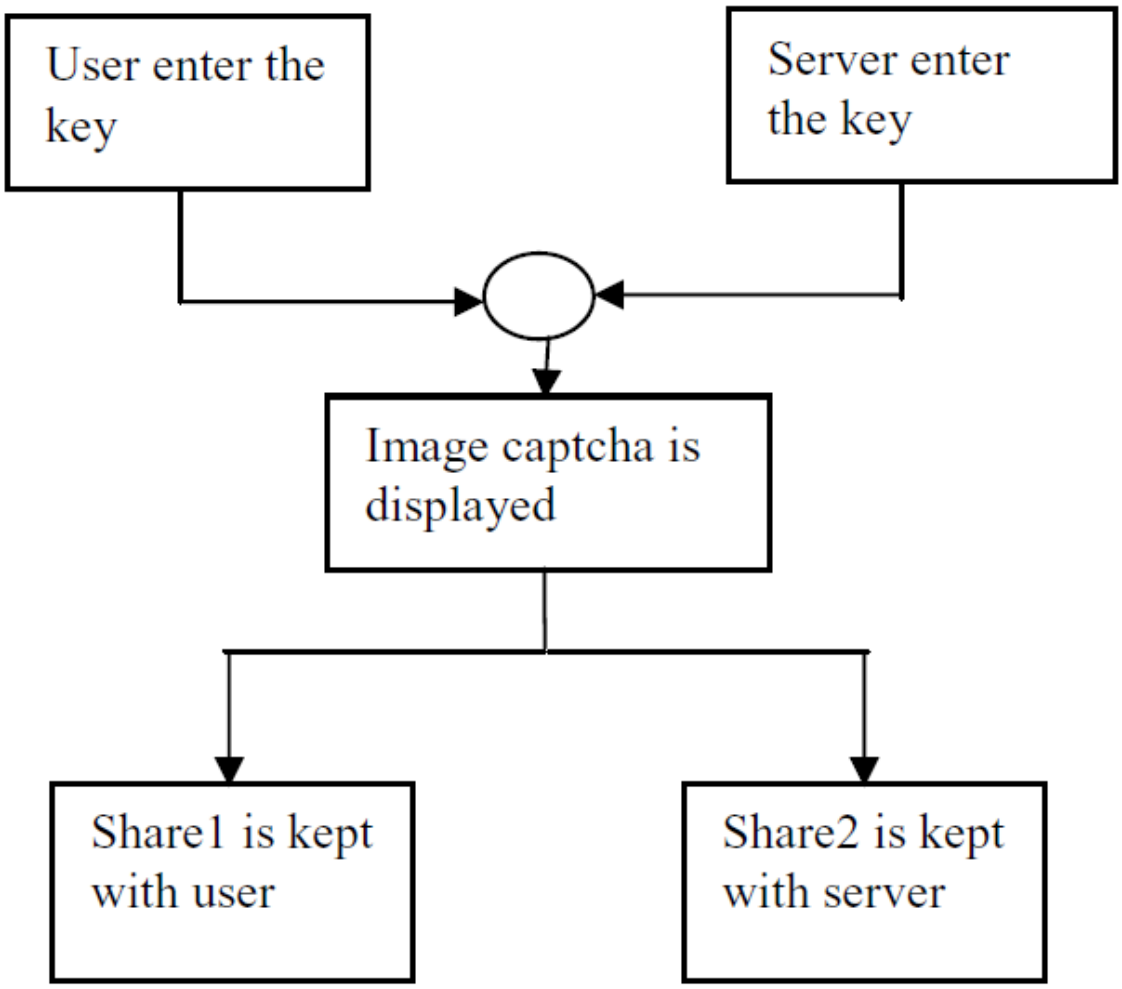

Figure 13: User registration process for the website.

This scheme protects the user account against phishing attacks if the connection between the user and the server is encrypted using SSL/TLS. However, a usability and accessibility problem could arise for the user as he is required to upload his share of the image each time he wants to login to the web application; a logistics problem would arise as the image share should be available on the computer from which the login process will take place.

3.2.1.2 A Strong Authentication Protocol based on Portable One–Time Dynamic URLs

Another authentication scheme proposed by E. Gal´an (E. GAL´AN et al., 2010), "A Strong Authentication Protocol based on Portable One–Time Dynamic URLs", this scheme relies on generating one-time dynamic and portable URL for each user once he

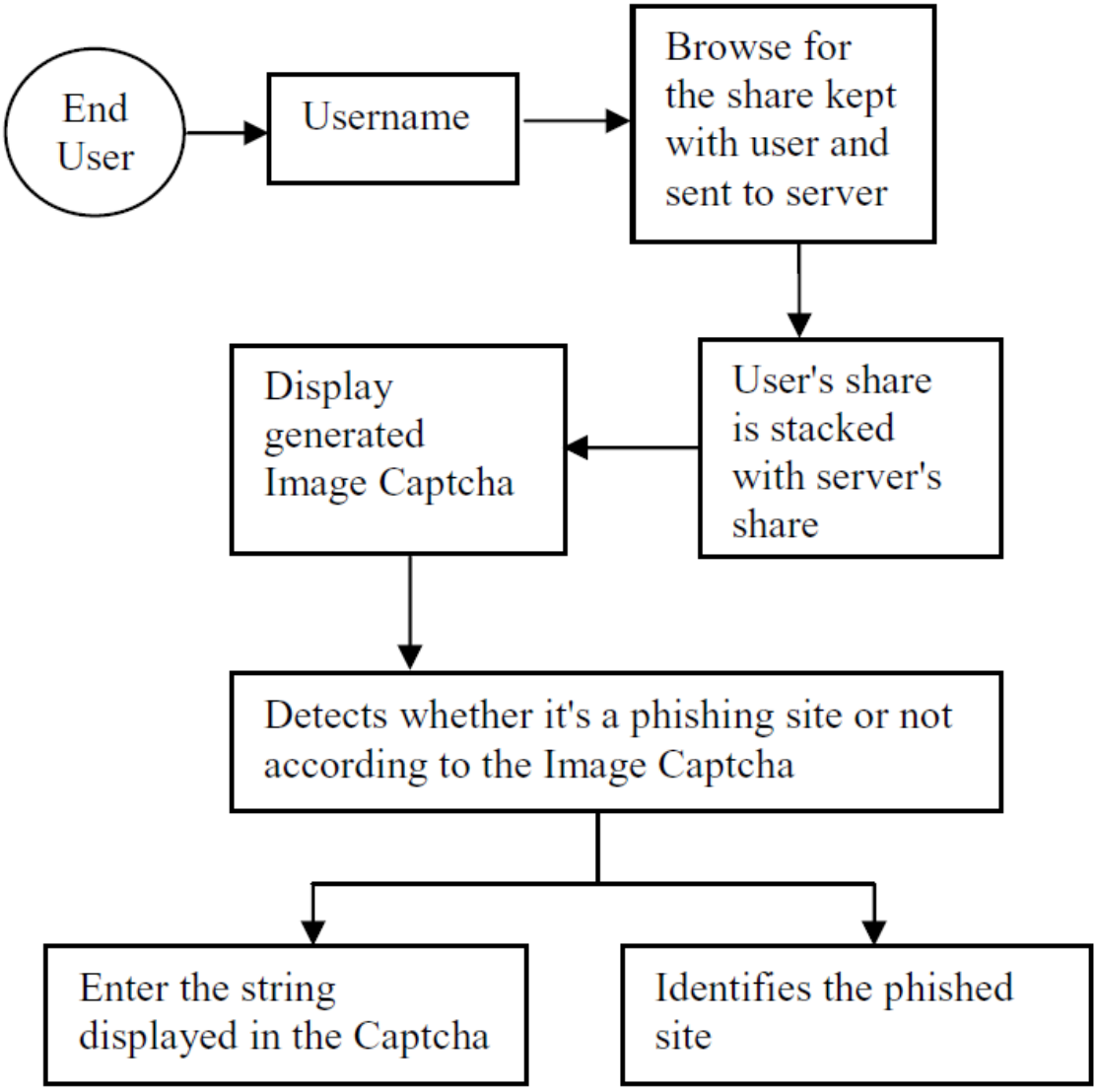

Figure 14: user logon process for the website.

logs in to the web application. This URL is generated specifically for the user in the specified session, and then sent to the user through a predefined communication channel (like SMS or email address).

After generating the URL, the server encrypts it using a shared key with the user; when the user receives the encrypted URL he is required to decrypt it and then access the web application through this URL.

The steps of the registration phase are detailed below:

1. User name and password are defined.
2. A private profile is created for the user U within Sapp with U's private data.
3. U specifies an additional channel, denoted as C, for communication. For example, an email address or a mobile number.
4. Finally, U and SApp agree upon a secret key KU.

The steps of the login phase are summarized below:

1. U submits user name and password via web.
2. SApp verifies the login and password submitted.
3. SApp generates a new portable onetime dynamic (POD) URL that points to a site that contains all data from the user's personal profile and some other functionality.
4. SApp stores the POD–URL just generated together with the session identifier.
5. SApp cyphers POD–URL with KU denoted as {POD–URL}KU .
6. SApp sends {POD–URL} KU through channel C (C is specified in the registration phase).

While this scheme prevents phishing attacks; it has a usability and cost weaknesses; the website vendor needs to send the encrypted one-time URL to the user through a medium like SMS that is a paid service, in addition, the process of decrypting the received URL requires a software service to be maintained by the user for this task.

3.2.1.3 CamAuth

Xie et al. proposed CamAuth (MENGJUN XIE et al., 2015), a 2FA scheme that leverages user's mobile as a second authentication factor, where user identity is proved using a

combination of Diffie-Helman keys exchanged between the client browser (through an extension or Add-on) and the server, and then verified using the user's mobile device via exploiting both the user PC and mobile cameras to exchange data that is encapsulated within a QR code.

Figure 15 depicts CamAuth authentication overview

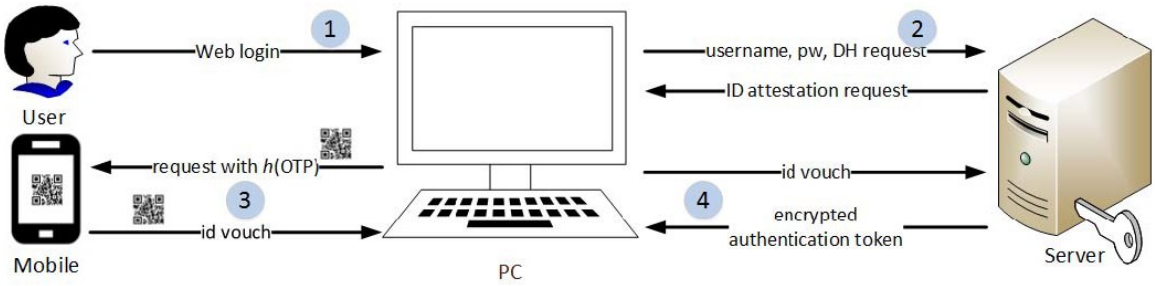

Figure 15: Overview of CamAuth authentication process.

Despite the security features that are implemented in CamAuth, three usability and deployability drawbacks could limit the adoption of such an authentication scheme:

1. Users are required to install a browser plugin to be used as a part of the authentication process; this requirement will limit the user who wishes to access his account from public computers.
2. CamAuth assumes that the user PC is equipped with a camera to be used as a medium to exchange data with the user mobile. This assumption is not true for a wide range of users whom PCs are not equipped with cameras, in addition to limiting the opportunities of users wishing to access their accounts from public or work computers.
3. The process of authentication and specially reading QR codes with both the PC's and mobile's cameras could result in usability inefficiencies and inconvenience for users.

3.2.1.4 Google Authenticator

Another category of 2FA schemes rely upon client side generation of one time passwords to be used as a second authentication token; a popular 2FA method that falls in this category is Google Authenticator (GOOGLE 2 STEP VERIFICATION). Google Authenticator (GA) is a mobile software that generates offline authentication codes that are used as a second authentication token; such that when the user access his account, he is requested to enter the generated code in addition to his credentials. GA generates authentication codes based on pre-shared secrets that were fed to the software in the registration process; they usually include user specific account details, code generation method (counter based or timestamp based), OTP characteristics, Etc. those pre-shared secrets are fed to the GA software through a QR code scanned with the user mobile camera. Figure 16 depicts a screenshot for GA.

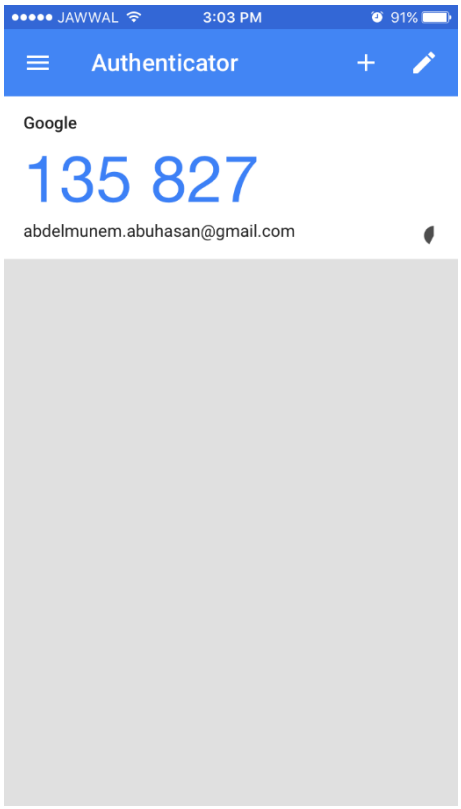

Figure 16: Screen shot for Google Authenticator.

Dmitrienko et al. (ALEXANDRA DMITRIENKO et al., 2014) performed a security analysis that concluded that such schemes are vulnerable to attacks especially in the registration phase. A PC standing malware can intercept the QR code that encapsulates the pre-shared secrets, and then the attacker can initialize his own version of GA and thus being able to generate valid authentication codes for the compromised account.

3.2.1.5 PhoneAuth

(A. CZESKIS et al., 2012) proposed PhoneAuth, a 2FA scheme in which the user mobile is considered a second authentication factor in addition to the user credentials; the user is authenticated after signing the login ticket (generated by the server) with the client private key that resides in the user' mobile. The login ticket is communicated back and forth between the client browser and the mobile application through Bluetooth. Figure 17 depicts an overview of PhoneAuth.

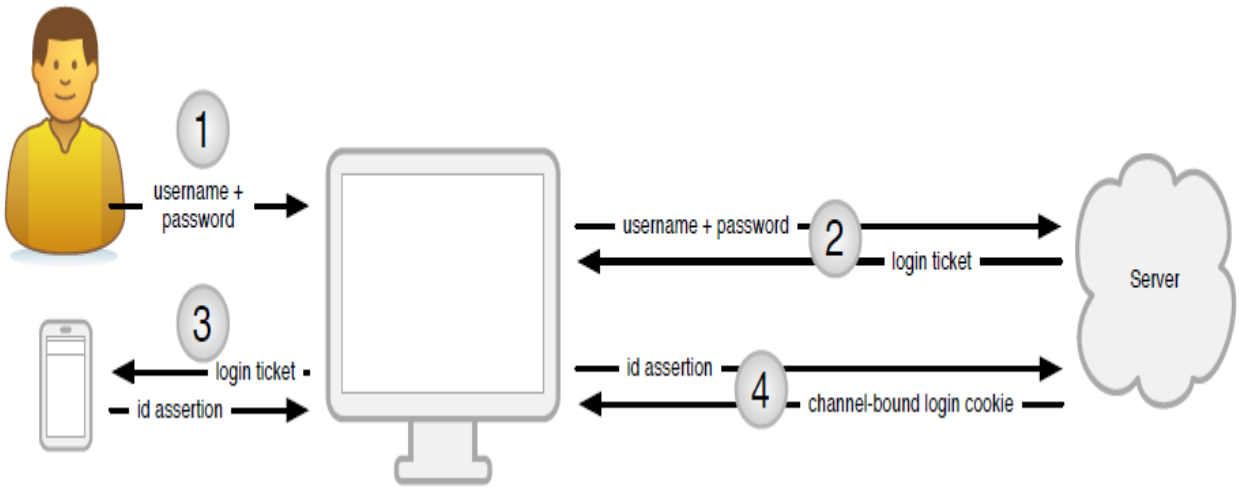

Figure 17: PhoneAuth Overview

PhoneAuth is built upon the origin-bound certificate, which modifies TLS to realize strong client authentication. The deployment of PhoneAuth requires modification to current TLS, web browser, and smartphone firmware, which is not practical for average users. Second, PhoneAuth relies on Bluetooth for communications between the smartphone and PC. However, Bluetooth can be subjected to a variety of attacks. The

Bluetooth module of smartphone has to stay active all the time, which is certainly not power efficient for mobile devices.

3.2.1.6 Snap2Pass

Ben Dodson et al. (DODSON et al., 2012) proposed Snap2Pass, a mobile based authentication system that aims at replacing the traditional password-based web authentication; leveraging either RSA model or symmetric key encryption.

The authentication process works as follows, see figure 18:

1. The user enters his user name on the browser and sends it to the server.
2. The server responds with an encrypted challenge (code) that is augmented on a QR code and sent back to the client's browser.
3. The user needs to scan the QR code through his dedicated mobile application and then decrypt and digitally sign the challenge with his private key or uses the shared key to decrypt the challenge and send it back to the server endpoint.
4. The server endpoint verifies the user response and authenticate the user session upon successful verification.

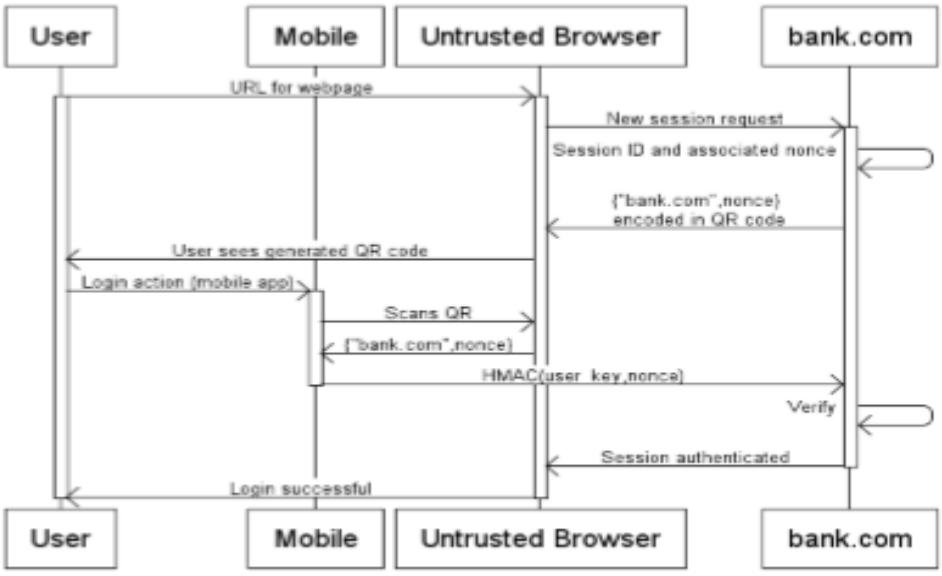

Figure 18: A sequence diagram for logging in to a web application using Snap2Pass

While this scheme successfully replaces traditional password-based web authentication, two weakness points could harm both the usability and security of this scheme:

1. It assumes that the user's mobile phone is connected to the internet to complete the login process.
2. In case of symmetric key approach, the shared key distribution is vulnerable to attacks that might enable a hacker to compromise the shared key and thus impersonate the user.

In chapter 5, we will work on resolving these two weakness points by enhancing this scheme to work in case the mobile phone is not connected to the internet and suggest a key distribution mechanism based on Diffie-Hellman key exchange protocol.

3.2.1.7 Phoolproof

Phoolproof (B. PARNO et al., 2006) exploits a trusted device (mobile phone) to perform mutual authentication that eliminates reliance on user behavior, defend against Man-in-the-Middle attacks, and protects user account from phishing, key loggers and most forms of spyware.

In Phoolproof, the user keeps a white list of the websites he visited before. When the user wants to visit a website, he selects the web site from the whitelist on his mobile phone, as shown in figure 19, and the phone communicates wirelessly with the browser to open the desired site to enter his credentials, then an end to end TLS mutual authentication takes place between the user's mobile phone and the website.

3.2.1.8 A Two-Factor Authentication System with QR Codes for Web and Mobile Applications

The study in (METE EMINAGAOGLU et al., 2014) employs QR codes to develop an alternative authentication model to the well-known OPT model whenever strong two factor authentication with high level of security is needed. The model uses dynamic random QR codes as OTP mechanism instead of the numeric or alphanumeric digits, the

QR code is communicated to the user via either MMS or email instead of relying only on SMS, which is the standard OTP mechanism today.

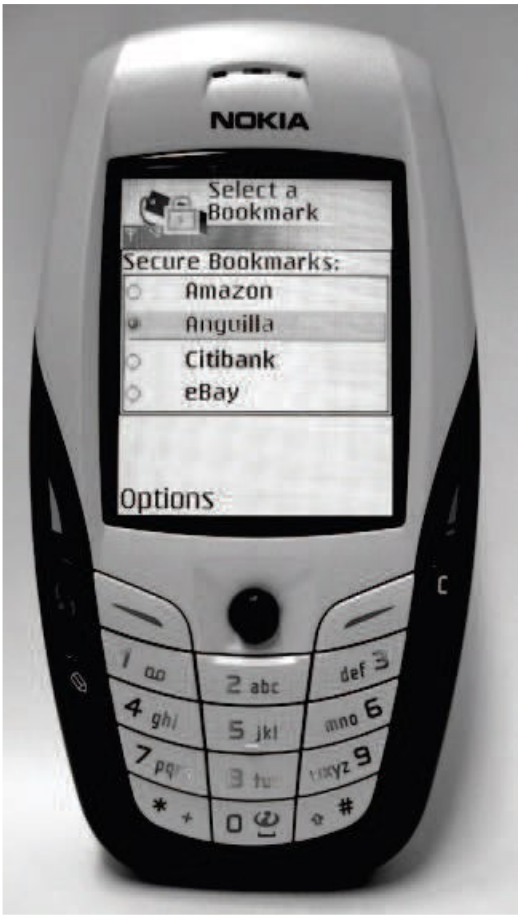

Figure 19: Cellphone User Interface in Phoolproof.

The basic idea of OTP authentication models is receiving randomly generated numeric/alphanumeric OTP data via SMS on the user's mobile phone and then manually reading that data from the mobile phone and then entering that data on the web page. However, in the proposed model in (METE EMINAGAOGLU et al., 2014), the user only receives the randomly generated numeric/alphanumeric QR code as an QR image via email or MMS, and scans that image on the web camera manually. The UML use cases and activity diagrams of the proposed system are shown in figures 20 and 21.

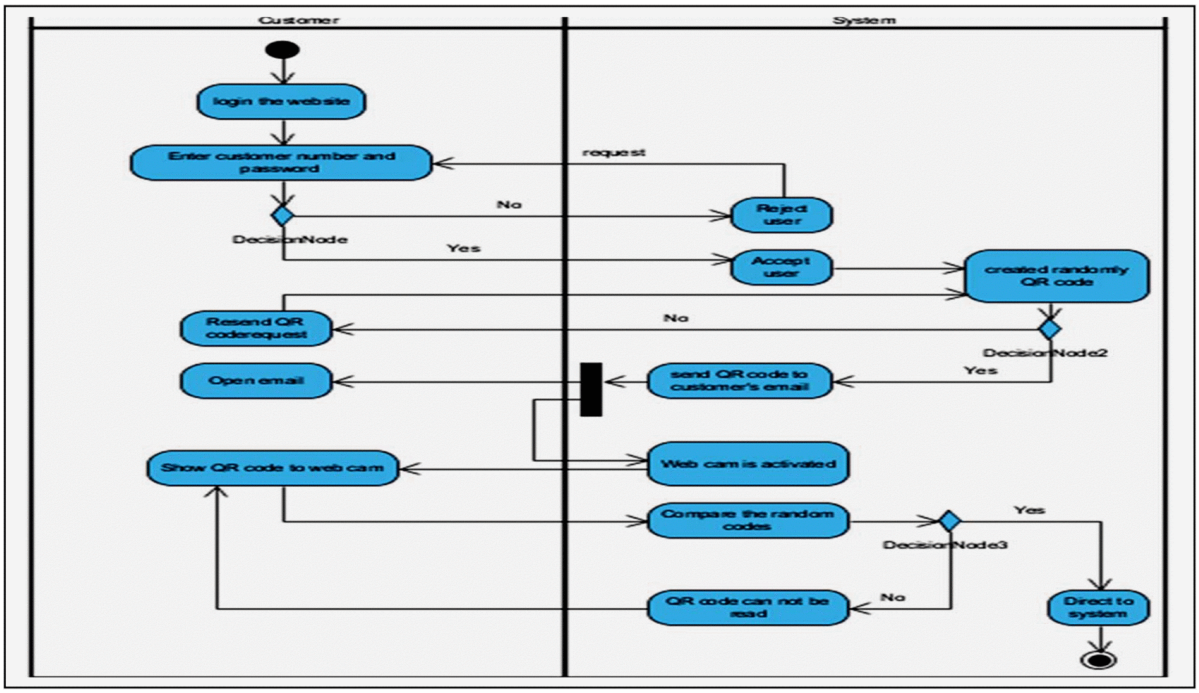

Figure 20: System activity diagram

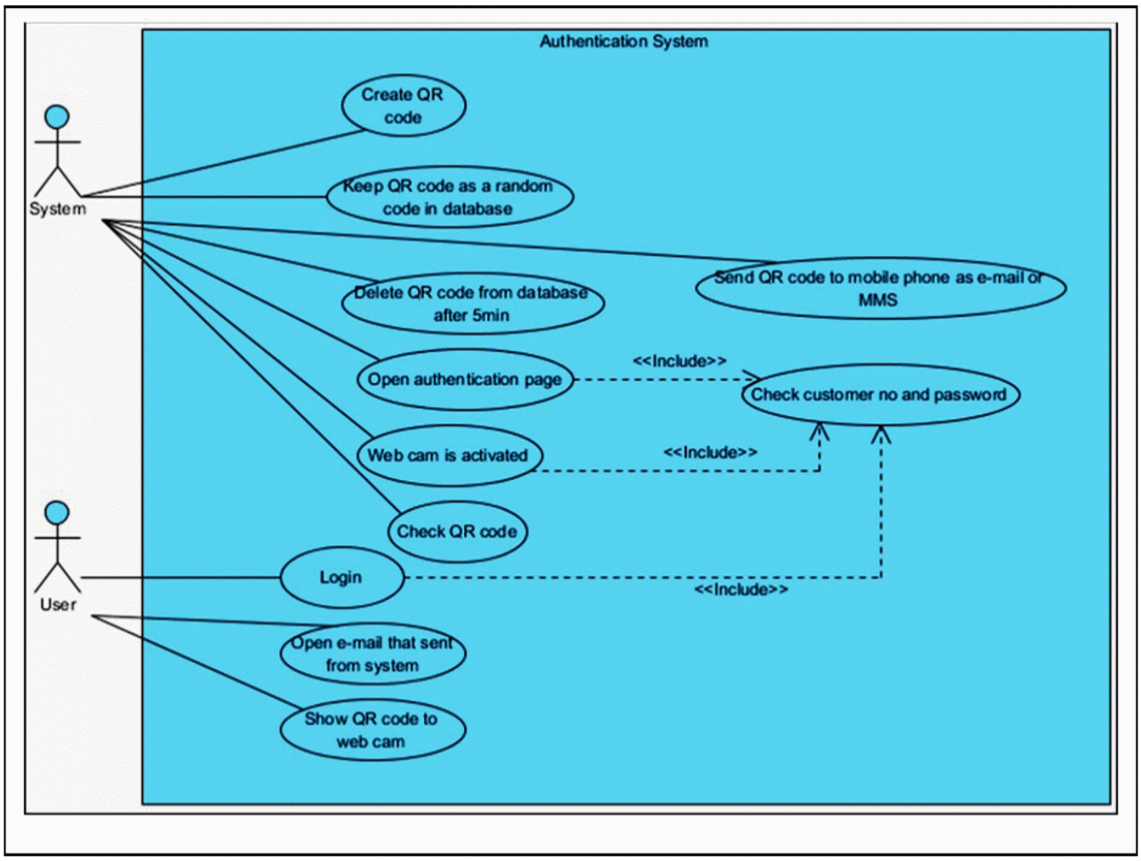

Figure 21: System use case diagram

3.2.1.9 QRP: An improved secure authentication method using QR codes

(DAVID PINTOR MAESTRE, 2012) proposed QRP, a 2FA authentication model that employs QR codes and the user mobile phone as a second authentication factor. In QRP, the user registration process involves the following steps:

- The user will submit her username, password and IMEI number of the phone he intends to use to authenticate.
- The web server will validate the entered data (correct IMEI, password complex enough, etc...), and then stores this information on the database.
- The server would generate a private and public pair of keys that are unique to the user, and then stores them on the server.
- After that, the user would proceed to download and install the application on his phone.
- The first time the mobile application is run, the user will need to enter her username and password (the IMEI can be verified by the mobile application) and the credentials (user/password) would be validated against the database through https request to the application server.
- If successful, three files would be imported and stored in the user's phone internal storage: the server's public key, the user's private key and a user data file, containing the user's encrypted credentials. The server's public key will be used to decrypt the credentials file. The user's private key will be used to authenticate in the server.

When the user opens the authentication page in the web application, a QR code containing a random number rn between 1 and 999999999 is presented to the user. Then the user needs to scan that QR code using his mobile application.

When the user opens the mobile application, he will need to input the password first. It will be verified against the user's encrypted file containing the credentials and if successful, the scanning application will run. The user can now proceed to scan the code from the web application screen.

The contents of the QR code will be captured and sent back to user mobile application, the mobile application will then generate a string containing the captured random number and the IMEI of the phone, that will be encrypted using user private key. Next, the mobile application will check the state of the phone and decide whether to authenticate the user in online or offline mode.

In the online authentication mode, the encrypted string of the random number and the IMEI of the phone in addition to the user name is sent back to the server through a secure https connection for validation, and upon successful validation, the server will authenticate the user browser session, as shown in figure 22.

In the offline mode of authentication where the phone detects that the Internet cannot be accessed, a unique six-digit number is derived from the encrypted string using an internal algorithm. This number is the pin code that the user will need to input in the authentication screen within the web application, along with his username. The pin code is entered through a screen keyboard, in order to avoid key loggers. Then server receives the username and pin code, recreates the pin code using the user's private key, the random number shown and the user's IMEI. If the pin code matches, then the server authenticates the user session. The offline mode of authentication is depicted in figure 23.

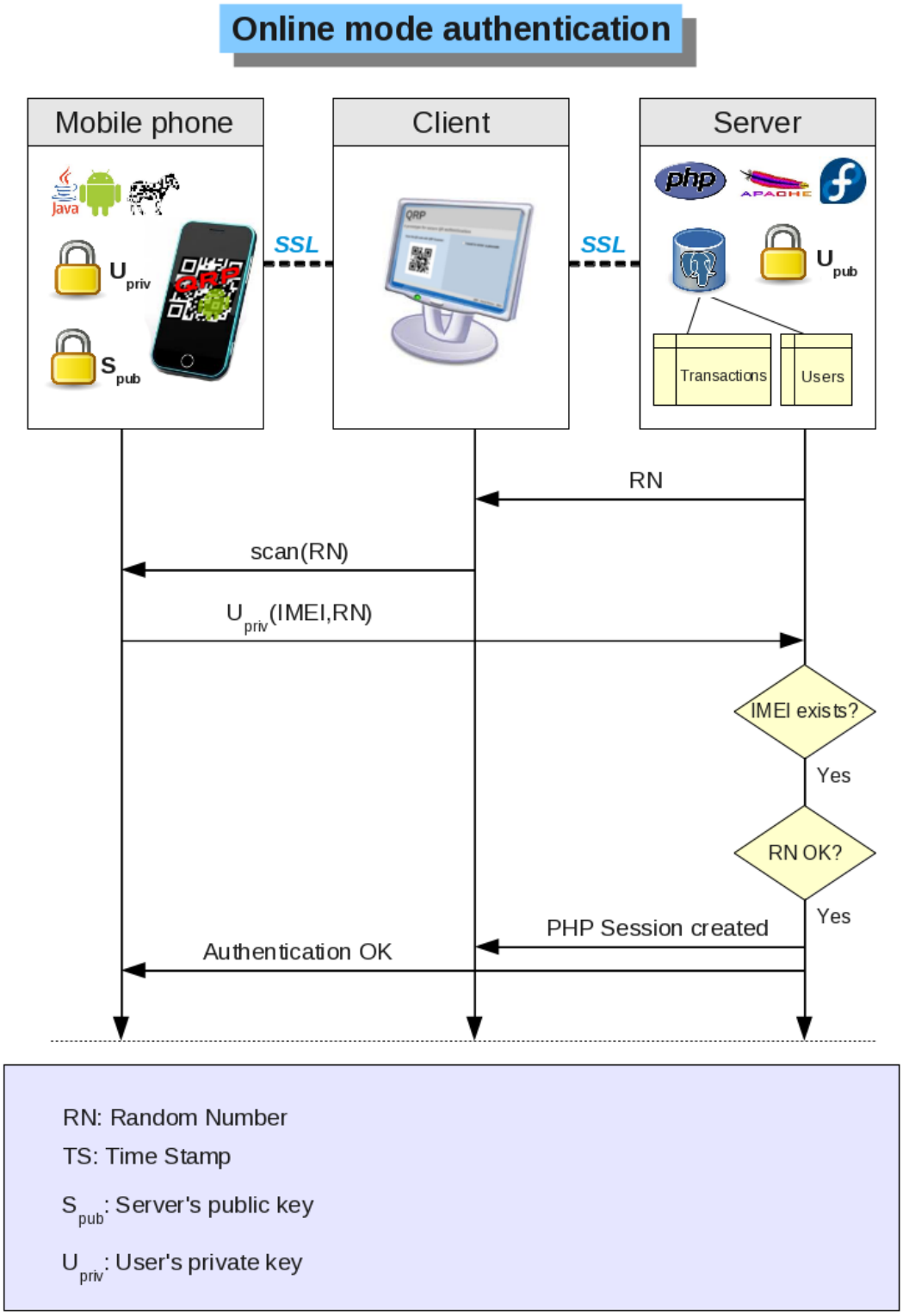

Figure 22: QRP online authentication mode

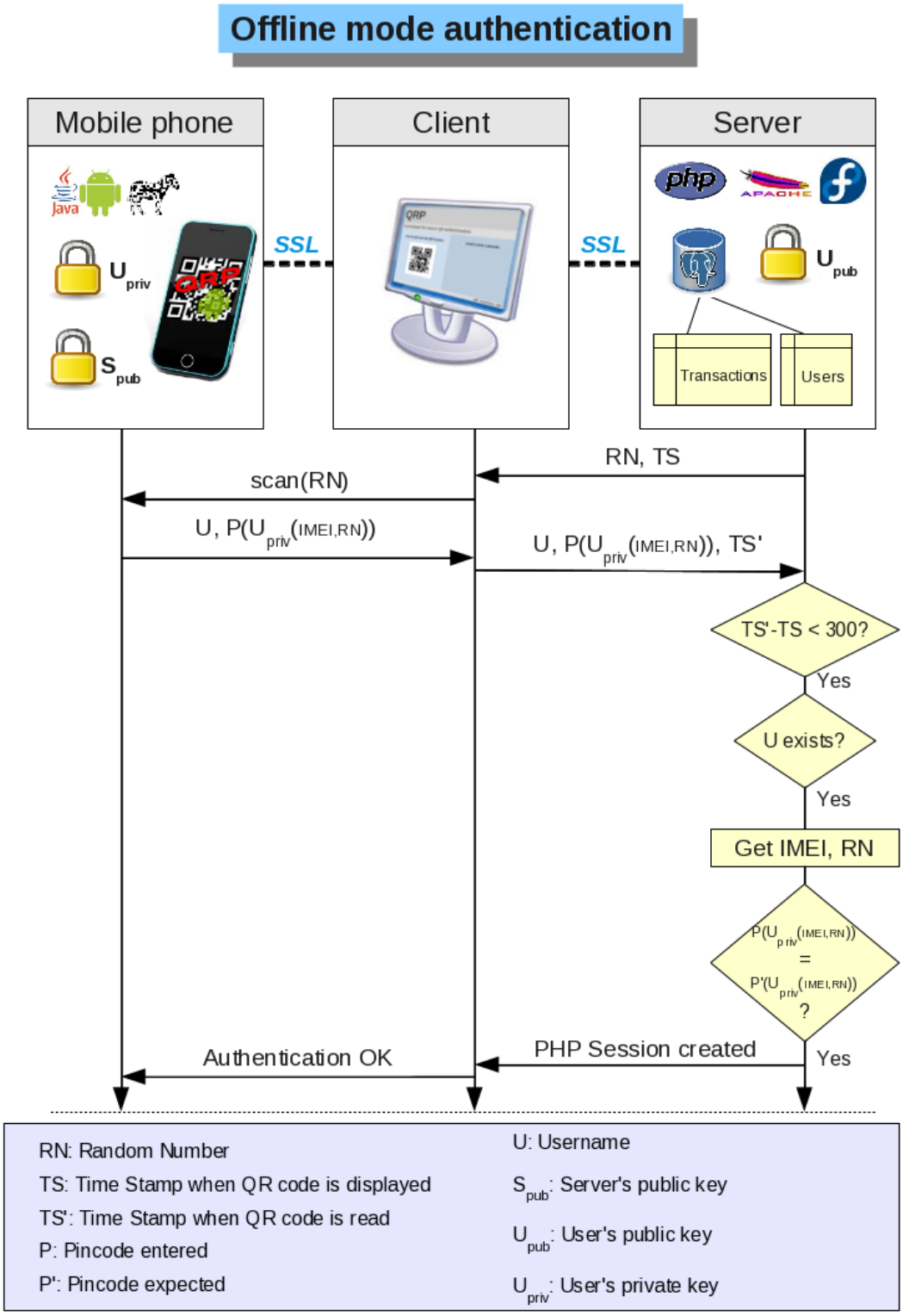

Figure 23: QRP offline authentication mode.

3.3 Summary and conclusions

Two-factor web authentication schemes employ their security enhancements to combat phishing attacks; adding another level of security to the authentication process makes it harder for an attacker to compromise a user account. The following design guidelines contributes to the success of any 2FA scheme in fighting phishing attacks in terms of security and usability features that makes it feasible to implement:

Prevention rather than detection: the key of success for any authentication scheme that aims at fighting phishing is to develop techniques to prevent phishing rather than relying on detecting phishing attacks and then preparing solutions to combat them.

Provide mutual authentication: as phishing attacks try to lure the user that he is visiting the legitimate site in his mind, it is very important to design authentication techniques such that clients should have strong guarantees that they are communicating with the intended recipient, and servers should have similarly strong guarantees that the client requesting service has a legitimate claim to the accounts he attempts to access.

Reduce reliance on users: it is very important to design authentication schemes that moves the burden of classifying or detecting phishing content from end user, it is the web application design responsibility to protect the user and the web site from potential attacks.

4 Combating phishing attacks with multi-level authentication

4.1 Design Overview

In this chapter, we propose a new authentication scheme that is considered a realistic solution to phishing attacks; this scheme moves the burden of securing user accounts from the users themselves towards the authentication system of the target web site, thus eliminating the potentially vulnerable schemes that depends on the user behavior against phishing attacks. This scheme enables the user to control and confirm every login attempt into his account by exploiting a set of smart trends in the web technology and digital data security. It combines the PKI robustness in data encryption with the prominent Google Cloud Messaging services (GOOGLE CLOUD MESSAGING) to tie up the authentication process with the user's ubiquitous mobile device in a secure, usable and realistically deployable authentication model.

The proposed authentication technique, which is called EARMAT (in reference to Enhancing Anti-phishing with Robust Multi-level Authentication Technique), aims at bridging the gap between current anti-phishing or phishing detection techniques and the continuous adaptation of attackers to those solutions. EARMAT goes beyond current anti-phishing techniques by enabling the user to make clear and intuitive decisions to allow or deny any login attempt to his account without the need of any security awareness level. To achieve this goal, EARMAT has been built to meet the three competing properties of any successful authentication scheme: security, usability and deployability.

In order to meet the goal of providing secure, usable, deployable and phishing-resistant authentication system, EARMAT has been built to achieve the following requirements:

1. EARMAT applies the principle of mutual authentication to eliminate replay attacks, Man in the middle attacks and phishing attacks.

2. EARMAT employs the user's smart phone as a second authentication factor building on the assumption of ubiquitous nature of mobile phones for every user.
3. EARMAT involves minimal possible user interaction in the second authentication factor; all the user needs to do is clicking a button on his mobile to confirm or reject the authentication attempt.
4. No changes are required on the user's PC or mobile phone for the authentication protocol to work, all the security features and communications of the authentication system are built over built in features in authentication parties; i.e. Browser, mobile phone and web server.
5. No operational costs are needed on the web application vendor or the user, as the communication between the web application and the user's mobile is initiated through the free Google Cloud Messaging service (or any other free cloud messaging service, e.g. Pushy (PUSHY)).
6. The user's mobile is assumed to have internet access to complete the authentication process; we assume that the clients of a website that implements an advanced authentication scheme like EARMAT are supposed to be connected to the internet all the time.
7. The protocol implements a fallback mechanism to enable the user to access his account in case of being not able to complete the authentication process using his mobile phone, or in case the GCM notification service fails.

Figure 24 depicts the general model of EARMAT authentication process. Once the user submits his credentials, they are validated in the web server and upon success the web server sends a login approval request notification message to the user's mobile device through GCM. Then the mobile displays the request details to the user who confirms or

rejects the login attempt, the user decision is sent back to the web server to authenticate or reject the login attempt.

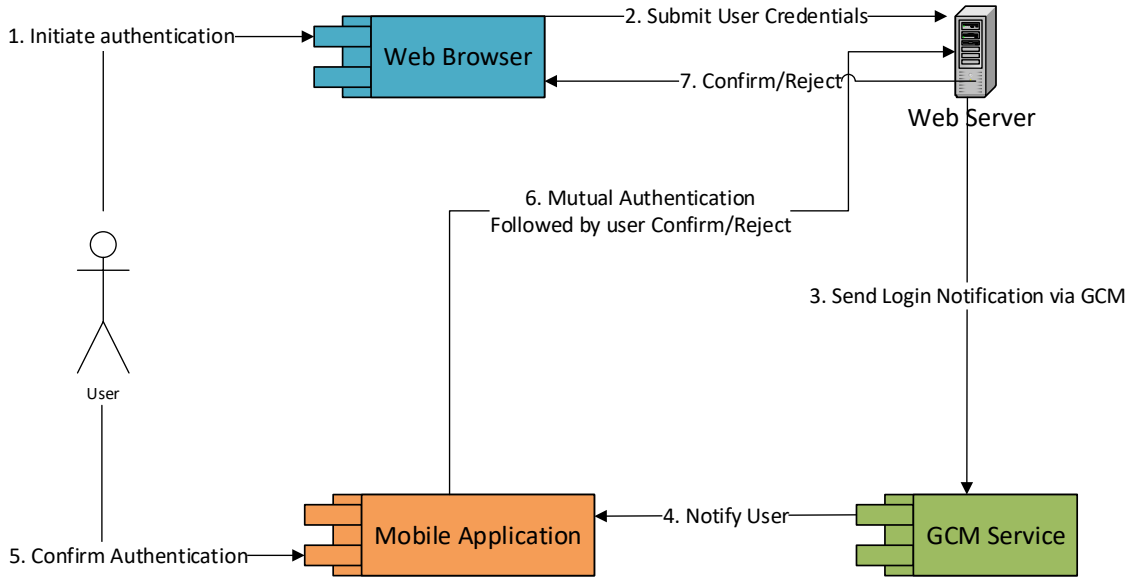

Figure 24: EARMAT Model Diagram

The basic building blocks of this authentication technique are:

1. The process of communication between the web server and the user's mobile application and vice versa through GCM.
2. The process of mutually authenticating the web server and user's mobile application to each other through digital signatures and PKI.

In the following sections, we discuss in details the basic building blocks of this authentication scheme in addition to evaluating it against a set of similar authentication schemes. Section 2 discusses Google Cloud Messaging (GCM) architecture, in section 3 we discuss the concept of mutual authentication and the PKI. In section 4 we discuss the details of the EARMAT protocol architecture, the protocol implementation and evaluation is discussed in section 5 followed by performance evaluation in section 6. Section 7 discusses the EARMAT fallback mechanism, section 8 introduces the protocol

management, the evaluation of the protocol and its evaluation analysis is discussed in sections 9 and 10 followed by protocol summary and conclusions in section 11.

4.2 Google Cloud Messaging (GCM)

GCM is a free service from Google that allows developers to send messages between servers and client applications and vice versa (GOOGLE CLOUD MESSAGING). GCM messages can carry up to 4Kb of payload, which makes it a feasible medium to carry notifications or small-sized messages from web servers to mobile applications; such as notifying an application that there is a new data to be fetched from the server. A GCM implementation includes a GCM connection server, an application server in the development environment that interacts with the connection server via HTTP or XMPP (Extensible Messaging and Presence Protocol), and a client application, as shown in figure 25.

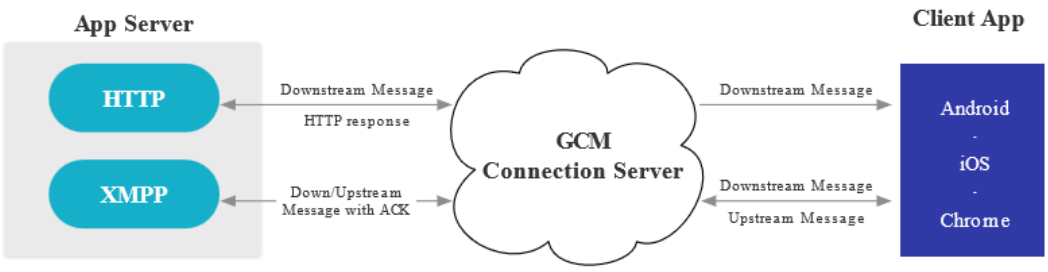

Figure 25: GCM Architecture

The GCM connection server accepts downstream messages from the application server and sends them to the client application; it is also possible to send upstream messages from the client app to the server through the GCM connection server using XMPP. The developer needs to implement HTTP and XMPP protocols in the App server to send downstream and upstream messages respectively through the GCM connection server.

To enable and use GCM for both the app server and the client app, the developer needs to register a new project through the Google's developer console, and gets associated with that Project ID. In order for the client app to receive/send messages through the GCM connection server, it must be GCM-enabled; it should be registered with GCM and get a unique identifier (Registration Token) that is associated with the app server project ID so that the app server can send notifications to specific clients through their own Registration Token. This collaboration framework is shown in figure 26.

EARMAT exploits GCM to send login notification requests when the user tries to login to his account, where the app server sends the notification message and waits for the user confirmation after being mutually authenticated with the user's mobile app.

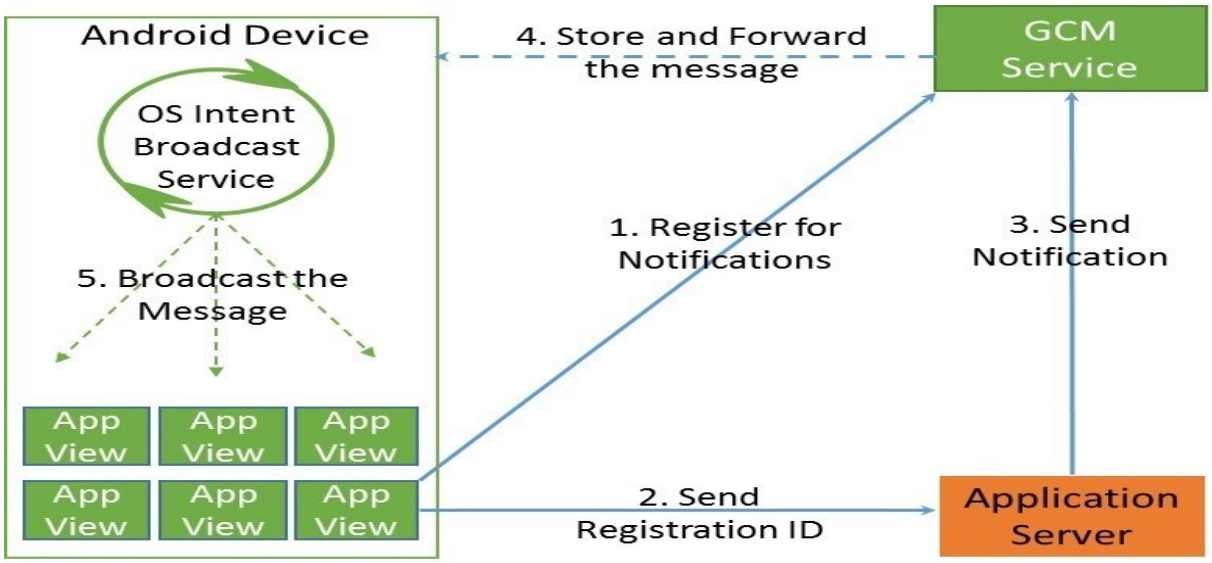

Figure 26: GCM Collaboration Framework

4.3 Mutual Authentication

Mutual authentication can be simply defined as a system security feature in which a client must prove its identity to the server and the server must prove its identity to the client before any process data get communicated between them. In the context of web authentication, the web server needs to make sure that the client is actually who he

claims to be and the client also needs to make sure that the server is actually who he claims to be.

Entities prove their identity through the usage of digital certificates; an electronic "passport" that allows a person, computer or organization to exchange information securely over the Internet using the public key infrastructure (PKI). Using this certificate, other parties communicating with that entity make sure of its identity. To provide evidence that a certificate is genuine and valid, it is digitally signed by a root certificate belonging to a trusted certificate authority (CA). Operating systems and browsers maintain lists of trusted CA root certificates so they can easily verify certificates that the CAs have issued and signed

Implementing mutual authentication in web applications adds an extra round trip time for client certificate (identity) to be sent back to the server after validating the server's identity. This extra step and its logistics, cost and complexity requirements are the main reason why most web applications are designed to not implement mutual authentication. However, the security threats that target websites (especially financial sites) -including phishing attacks- make it necessary to balance between the mutual authentication complexity and the security features it offers.

EARMAT implements mutual authentication between the app server and the client's mobile app using RSA based PKI to protect the user from online fraud such as phishing, man in the middle attacks and pharming attacks. In this context, we explain simply how the RSA encryption works.

RSA is asymmetric cryptographic algorithm that is used to encrypt and decrypt messages, also known as Public Key Cryptography and includes two keys: a public

key that is assumed to be known to everyone and private key that is known only to its owner.

In RSA cryptography, messages encrypted with one key are only decrypted with the other key; this feature leads to two important usage scenarios:

1. Any entity can send encrypted messages (with the public key) to the server, and only the server can decrypt those encrypted messages as he owns the private key.
2. The server can digitally sign any message that he sends to any party with the server's private key, which guarantees to the receiver (who knows the server's public key) that the message was issued by the server.

The second scenario above forms the basis of mutual authentication that is implemented in our authentication technique. In order to prepare the public and private keys of an entity, the following algorithm takes place:

1. Choose two different large random prime numbers p and q.
2. Calculate $n = pq$, n is the modulus for the public key and the private keys.
3. Calculate the totient: $\phi(n) = (p-1)(q-1)$.
4. Choose an integer e such that $1 < e < \phi(n)$, and e is coprime to $\phi(n)$ **i.e.:** e and $\phi(n)$ share no factors other than 1; gcd(e,$\phi(n)$) = 1.
5. e is released as the public key exponent.
6. Compute d to satisfy the congruence relation $de \equiv 1 \pmod{\phi(n)}$ **i.e.:** $de = 1 + k\phi(n)$ for some integer k.
7. d is kept as the private key exponent.

4.4 EARMAT protocol architecture

EARMAT achieves its design goals by extending the traditional password authentication scheme with a new level in which the user decides whether to confirm the authentication process or not, this new authentication level takes place through the user's mobile device through a dedicated mobile application; so that the app server sends a confirmation request to the user's mobile to complete the authentication process. Figure 27 outlines the basic steps involved in EARMAT authentication

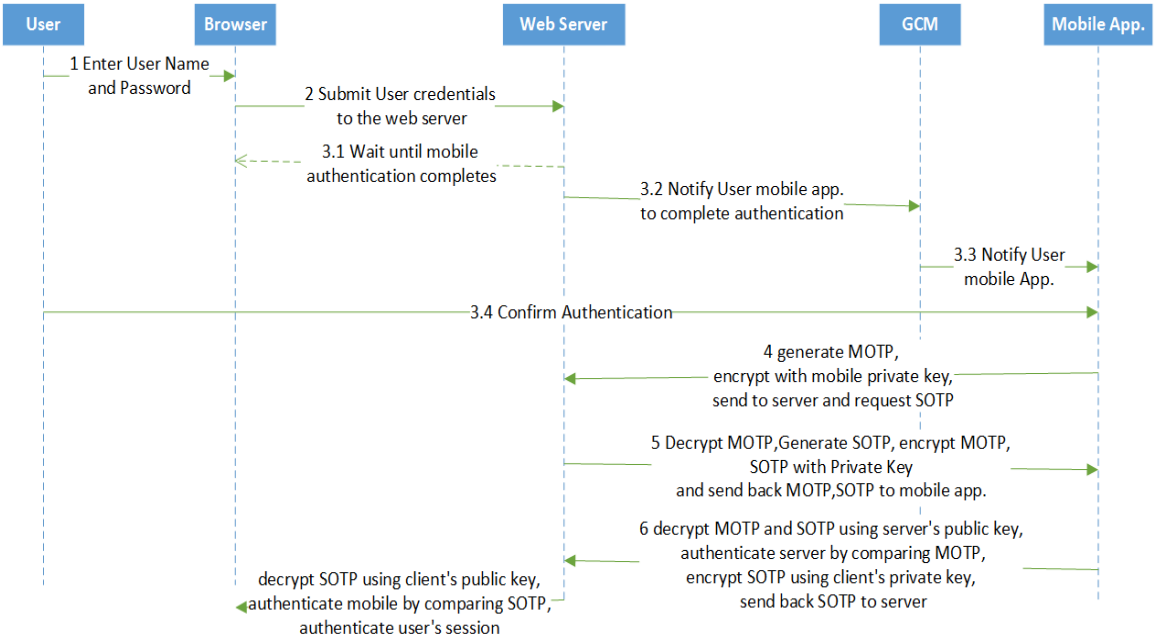

Figure 27: EARMAT Authentication Sequence Diagram

Based on the notations in table 2, the EARMAT authentication protocol steps are:

Step1: the user enters his credentials (U and PWD) via the web app login form, and then submits them to the server for verification.

Step2: the server verifies the received user's credentials. Upon negative verification, the login attempt fails and the user is notified through his browser. Upon success, the authentication protocol proceeds to step 3.

Step3: The server responds with a wait response for the client browser, informing him that he needs to complete the authentication process from his mobile device.

The server looks up the registration token (RegToken) associated with the user (explained later) from local database, and sends a login notification to the user's mobile application via GCM; GCM will deliver the notification message (includes LID) to the user's mobile application directly.

Table 2: EARMAT Notations

Notation	Description
U	User name
PWD	User password
LID	Login Attempt ID
MOTP	Mobile One-Time Password
SOTP	Server One-Time Password
RegToken	Mobile App. Registration Token in GCM
GCM	Google Cloud Messaging Service
PID	Project ID that identifies the web application in GCM
SPU	Server Public Key
SPR	Server Private Key
MPU	Mobile Public Key
MPR	Mobile Private Key
NLR	Negative Login Result
IMEI	User mobile phone International Mobile Equipment Identity which is used to uniquely identify a mobile device.
NMSG	Notification Message

Step4: once the login notification message is received by the user's mobile application, the application will show a confirmation dialog so that the user decides whether to allow and complete the authentication process or not. If the user rejects the login attempt, an encrypted (with MPR) negative response message (NLR) is sent back to the server over

a secure connection (SSL) channel indicating the failure of the authentication, then the server verifies the response and responds to the user browser with a failure of authentication message. If the user accepts the login attempt, the protocol proceeds to step 5.

Step5: the mobile application generates a random code (MOTP), encrypt it with its private key (MPR) and sends a request to the server containing the encrypted MOTP, IMEI and LID to get the authentication token.

Step6: the server decrypts and encodes the received token, generates a OTP associated with the current login session (SOTP), specify its validity period, combine them as an authentication token, stores it locally, encrypts it using the server's private key (SPR) and sends back the encrypted authentication token to the client mobile application over a secure channel.

Step7: the mobile application receives the encrypted authentication token, decrypts it using the server's public key (SPU), decode the result and compare it with previously sent MOTP Then checks its validity and encrypt it using its own private key and sends it back to the server over a secure channel.

Step8: the server decrypts the received authentication token using the client's public key, verifies it and compares it with the previously stored OTP (SOTP) for the current session id. If the two OTPs match, the server authenticates the current user session and notifies the user's browser to be redirected to the website main page.

Assumptions and notes

The following assumptions are made to supplement the general architecture of EARMAT authentication protocol:

1. The protocol relies on GCM for notifying the user's mobile application of the current login session so that the user completes the authentication steps.
2. User's sensitive data are never communicated through GCM: EARMAT exploits GCM as a notification service to enhance the user convenience and improve usability features in the authentication protocol. GCM is not considered a proprietary third party for two reasons: other free notification services could be used as well (e.g. Pushy) and an alternative option is to adopt the design of enabling the user himself to initiate the process of completing the authentication using the mobile application by pulling pending authentication requests from the app server.
3. EARMAT implements a fall-back mechanism to enable the user to bypass it in case his mobile is not accessible at the time of login.
4. The generated authentication token is session-specific; i.e. it is valid for the current user login session only, in case an attacker compromises it, it will be no longer valid for any other session initiated by the attacker.

4.5 EARMAT Implementation and Evaluation

4.5.1 Registration Phase

EARMAT activation process includes a set of configuration steps that are needed from both the server application and the mobile client application; in order for any web application to use GCM, it should be registered with a GCM connection server from google.

4.5.1.1 Server Registration with GCM

The web app admin needs to create a new API key from google developer console (GOOGLE DEVELOPER CONSOLE); this API key will be used by GCM to identify the web application and manage its activities such as sending messages. The API key

creation is a simple process that is initiated from the Google developer console, as shown in figures 28 and 29.

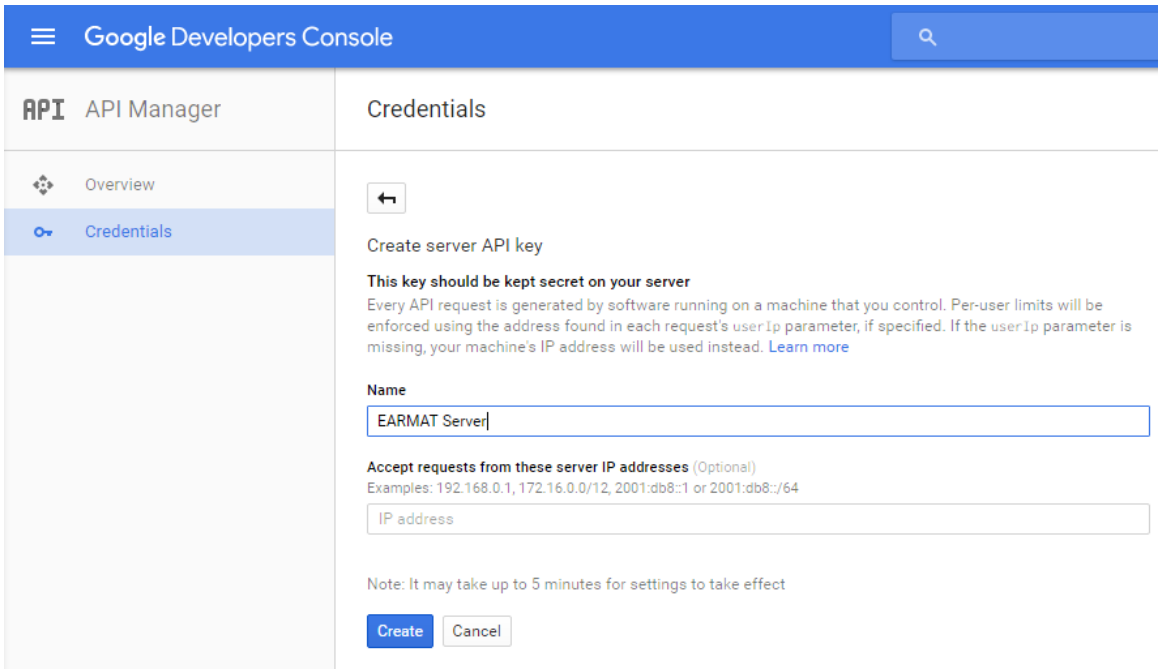

Figure 28: API key creation in Google Developer Console

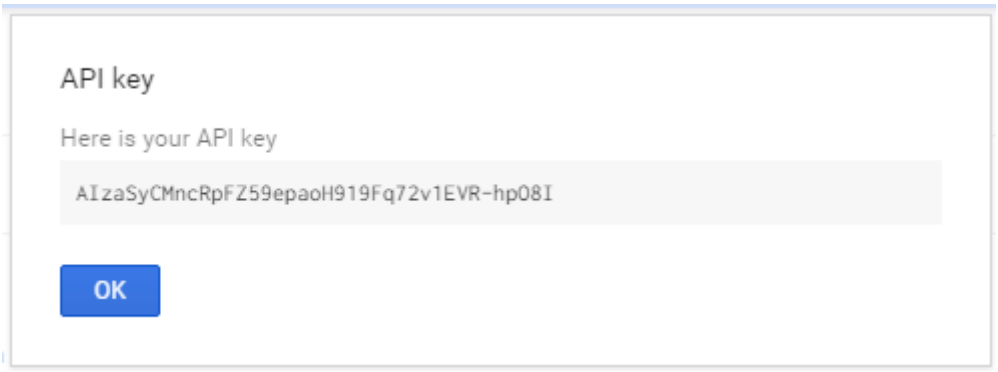

Figure 29: Sample API key

After generating the API key, the web application needs to store it locally and never share it with any other parties; as it will be used as an identifier of the web application when communicating with GCM connection server to send messages.

4.5.1.2 Server RSA keys generation

The web application needs to generate an RSA key pair (private and public keys); the private key is stored securely in the server's key store; the public key is distributed to the clients in a secure key distribution mechanism. Those keys will be used to encrypt/decrypt messages sent and received by the server as part of the mutual authentication step with the user's mobile application as will be explained later.

4.5.1.3 Mobile app installation

At this step, we assume that the user has filled in his sign up information including his credentials (user name and password), basic information and specified his mobile IMEI (International Mobile Station Equipment Identity). After that, the user needs to download and install the mobile application as part of his registration process for the website, the mobile application could be available for download from the user's mobile platform app store or could be provided to the user at the end of the sign up process to be downloaded through a web link.

4.5.1.4 Mobile app registration with GCM

The user needs to login on the mobile application using his username/ password pair, In the first login to the mobile application, the application will register itself on GCM under the project id that was associated with the web application as explained earlier. Upon registering, GCM will generate a registration token associated with the mobile application who in turn will send it to the web application to be stored and paired with the user name to be used later for sending notifications from the web application to the user's mobile application.

4.5.1.5 Mobile app RSA keys generation

The mobile application needs to generate an RSA key pair (private and public keys); the private key is stored securely in the mobile key store; the public key is communicated to

the server in a secure key distribution mechanism. Those keys will be used to encrypt/decrypt message sent and received by the mobile app as part of the mutual authentication step with the server as will be explained later.

4.6 EARMAT Implementation

In order to assess its design requirements, we built a prototype system that implements the EARMAT authentication scheme, the prototype includes the two main components of the system: a web application and mobile application, in addition to implementing the authentication steps and the required configurations including key generation and communication.

The web application was built using Java platform, namely using JSF (Java Server Faces) and Servlets framework. The web application offers a set of services for handling the following tasks:

1. Server registration with GCM: in this task the web application manages and configures the GCM API key of the project to be used later to send notification messages for the client's mobile. The server needs to store the API key in a secure and manageable manner.
2. Managing user accounts: the web server needs to implement a service to manage user profiles including credentials and GCM registration token for each user.
3. Sending notification messages to clients: the web app needs to handle the process of sending notification messages to clients through the GCM server using the API key and the user's registration token as identifiers for both the web app and the client respectively.
4. RSA keys generation and management module: the web app needs a service to manage the process of generating and maintaining the server's public and private keys.

5. User authentication module: the web app implements a service to manage user authentications including credentials verification and the process of generating, encrypting and communicating the authentication token to the client.

The client's mobile application is developed on android 5.1, and it is compatible with android 2.2 and upward platforms as no special APIs are used except for support to GCM (ANDROID CLIENT). The mobile application is responsible for device registration, confirming and completing the authentication process, the application uses the necessary APIs for RSA key generation, storage, encryption, decryption and communications with the server over SSL.

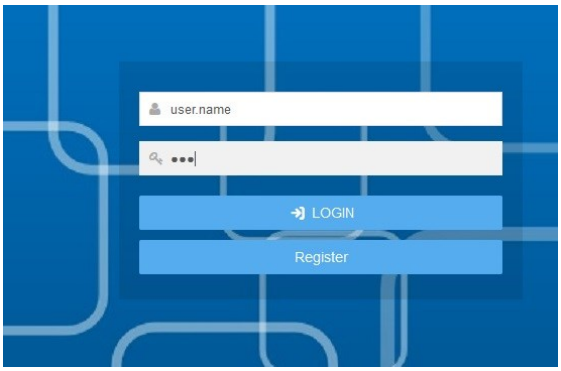

Figure 30: website login page

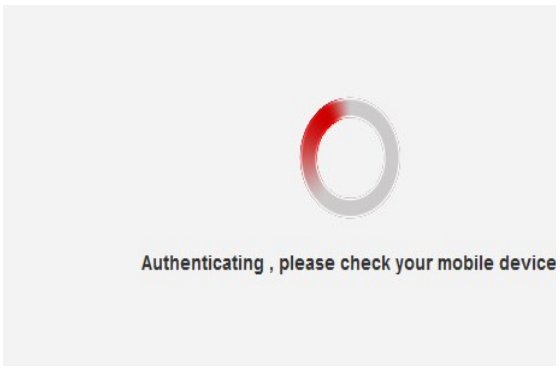

Figure 31: User authentication waiting message

Figures 30, 31 and 32 show screenshots of the EARMAT authentication process; figure 30 shows the website login page where the user enters his credentials and submit them for validation in the web application; upon successful validation, the server sends back a wait message to the user's browser (figure 31) and proceeds to generate the authentication token and sends it to the users mobile through GCM.

figure 32 (a) shows the login notification that appears on the user's mobile application upon the arrival of the notification message, the notification message displays to the user the login attempt details including the user name, login time, user OS and browser name,

so that the notification message gives the user a complete and clear details to enable him to decide whether to authenticate the session or reject it. The user decision is sent back to the web app through a secure communication channel and then the web app either authenticates the user login session or rejects it based on the user feedback.

Figure 32:(a) Login Notification. (b) Manual login completion if GCM notification fails

The prototype web application and the mobile application were deployed on a test environment; we used Glassfish application server to deploy the web application while the mobile application was tested using the mobile emulator of Android Studio. The system functionality has been tested and the authentication scenario worked successfully.

To assess the performance of EARMAT, we conducted a performance test using an emulator in android studio with the following specifications: Device: Nexus 5, CPU: x86, RAM: 1.5 GB, Platform Version: Android 5.1 (lollipop). In this evaluation, we measured response time and memory usage of the EARMAT mobile app,

The results of the performance test showed that EARMAT -in average- spent 4.3 Milliseconds to complete the decryption and encryption processes, while consuming 0.06 MB of memory. The whole memory reserved by the application was 2.38 MB. Table 3 depicts the test results of 15 runs of the algorithm of decrypting and encrypting process using the emulator.

Table 3: Performance test of EARMAT in the emulator

Run #	Execution Time (ms)	CPU Usage	Memory Usage (MB)
1	5	11%	0.08
2	3	12%	0.06
3	3	12.5%	0.07
4	6	9.5%	0.06
5	6	14%	0.05
6	3	20%	0.05
7	3	9%	0.07
8	4	11%	0.07
9	7	15%	0.06
10	9	10%	0.06
11	4	18%	0.05
12	8	22%	0.06
13	4	25%	0.06
14	7	9%	0.06
15	3	23%	0.05

The complete authentication process will include also the time spent in communications between the web server and GCM server which in turn notifies the user's mobile application in an automated process that is expected to add a very little time fraction (in milliseconds).

These performance test results indicate that EARMAT implementation in most modern mobile devices will be feasible and the response time will be accepted by users, making it possible to adopt such an authentication scheme at a compromise of a couple of seconds latency.

4.7 GCM privacy considerations

EARMAT uses Google Cloud Messaging (GCM) only to inform the mobile application about incoming login requests, the app then communicate directly with the web server to complete the authentications steps. Neither contents nor details about the authentication process, user mobile number or user specific data will be transmitted via GCM.

The user can choose whether he would like to use EARMAT with GCM, this option could be added in the mobile app settings such that the user can poll authentication requests manually through a dedicated service on the app as depicted in figure 32 (b).

4.8 EARMAT Fall-back Mechanism

Although we assume that users will typically have their mobile phones all the time, it is necessary to strengthen EARMAT with an alternative (fallback) login method; EARMAT authentication protocol can fall back to user's credentials only scheme in case the user's mobile is not compatible or inaccessible at the time of authentication. To strengthen the traditional password based authentication, the fallback mechanism is supplemented by an SMS-based or email-based One Time Password (OTP) that is delivered to the user to enter it in addition to his username/password credentials.

The web application needs to treat login sessions in the fallback mechanism in a less privileges mode; for example, the user will have limited authorizations on critical services such as resetting user password, financial transactions, etc.

If GCM notification service fails, no notification message is received on the user mobile application to guide him through the authentication process. In this case, the user can initiate a request from his mobile application to query the available login attempts that are pending to be authenticated, and then continue the authentication process as usual, as depicted in Figure 30 (b).

4.9 EARMAT Protocol Management

4.9.1 Key Maintaining

in order to activate and apply the mutual authentication principle in EARMAT, both the server and the client need to possess an RSA private and public key pairs; the key pairs are generated using a standard platform packages such as the Java security package in case the applications are developed using the Java platform.

The second important issue is how and where to maintain the keys, especially the private key. The security of private keys is crucial for public key cryptosystems; anyone who can obtain a private key can use it to impersonate the original user during all communications and transactions. Therefore, private keys must be in the possession only of authorized users, and they must be protected from unauthorized use.

For software-based public key cryptography, as in our case, cryptography operations occur in the computers operating system memory. Attackers might be able to force buffer overflows or memory dumps to obtain private keys. Even if a private key is protected by encryption while it is in memory, obtaining the protected key is the first step in a potential attack to discover what the key is.

In addition, many cryptosystems also store private keys on local hard disks. An attacker with access to a computer might use low-level disk utilities to locate encrypted private keys on the hard disk and perform cryptanalysis to decipher the key. In general, the risk

of attacks on private keys is much lower when keys are stored on tamper resistant hardware devices such as smart cards.

Most platforms maintain keys in the key store - a repository of security certificates or keys - which is usually protected by a password known only to its creator and protected through the platform under which it operates, while the threats on software and hardware level could compromise the key store contents as discussed above, many platforms store the key store on real, dedicated, tamper-resistant hardware module which makes it more difficult for an attacker to compromise the store keys.

4.9.2 Maintaining secure communication channels

Protecting the user's private data while in transit between the browser and the web application or between the mobile application and the web application is considered a first defense line against attacks. We assume that all communications are carried out under secure communication channels using SSL protocol.

4.9.3 Trusting and revoking user mobile device(s)

EARMAT authentication scheme supports trusting more than one mobile device for the user account, the user needs a full privileged active session to register a new device and associate it with his account, the user adds the IMEI of the device and associate it with his account, install the mobile application on the new device, register the mobile application with GCM service and sends the registration token, device IMEI, device public key to the server for verification and acceptance.

In the multi device mode, when an authentication attempt is initiated from browser, all the trusted and linked mobile applications on all trusted devices are notified of the login

attempt through GCM group messaging service, then the user can confirm and complete the authentication process from any device.

Users may want to revoke a device and disassociate it from their account (in case it is stolen, changed …). The revocation process could be done either from a fully authorized web session or by requesting that from the website vendor.

4.10 EARMAT Evaluation

To evaluate EARMAT, we conducted a comparative study with other four popular authentication techniques; namely: Passwords, Google 2 step-verification, PhoneAuth (A. CZESKIS et al., 2012) and CamAuth (MENGJUN XIE et al., 2015). The comparative study analyses a set of factors for each authentication scheme compared to EARMAT, each factor contributes to the degree of usability, security or deployability features of its authentication scheme.

The benefits measured to assess EARMAT success and feasibility are based on the 25 security, usability and deployability features that were put by the authors in (J. BONNEAU et al., 2012). They define it as "a standard benchmark and framework allowing schemes to be rated across a common, broad spectrum of criteria chosen objectively for relevance in wide ranging scenarios, without hidden agenda". This evaluation model assesses the strength of a web authentication scheme through evaluating it against a predefined set of features, acting as building blocks for the scheme overall security, usability and deployability benefits.

4.10.1 Usability

In Software engineering, usability is defined as the degree to which a software can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use. In the context of web

authentication, the usability features that achieve an authentication technique are categorized into eight properties in the assessment model in (J. BONNEAU et al., 2012):

1. **Memorywise-Effortless**: users of a web application that implements the authentication scheme need not remember any secrets at all to access the web application through that authentication scheme. The authentication scheme is considered as **Quasi-Memorywise-Effortless** if users have to remember one secret for everything (as opposed to one per web application that implements the authentication scheme).
2. **Scalable-for-Users:** this property measures the scalability of the authentication scheme from the user's perspective; i.e. does the scheme adds any burden or extra efforts on the user in case he needs to access multiple accounts (tens or hundreds of them).
3. **Nothing-to-Carry**: Users do not need to carry an additional physical object (electronic device, mechanical key, piece of paper) to use the scheme. The scheme is considered **Quasi-Nothing-to-Carry** if the user needs a second object that he carries everywhere all the time anyway, such as their mobile phone, but not if it's his computer or tablet.
4. **Physically-Effortless**: this property is given to an authentication scheme that do not requires any physical efforts or prior knowledge from its users, including typing, scribbling or any other motions. The scheme is evaluated as **Quasi-Physically-Effortless** if the user's effort is limited to speaking, on the basis that even illiterate people find that natural to do.

5. **Easy-to-Learn:** an authentication scheme is considered to be easy to learn if users who have no prior knowledge on how to use it can learn it quickly without any difficulties.
6. **Efficient-to-Use:** The time the user must spend for each login attempt is acceptably short, and the time required for setting up a new account with the web application that implements the authentication scheme is reasonable.
7. **Infrequent-Errors:** the authentication technique works properly; i.e. the login task that users must perform to access their accounts usually succeeds when performed legitimately. In other words, the scheme isn't so hard to use or unreliable that genuine users are routinely rejected.
8. **Easy-Recovery-from-Loss:** the authentication scheme needs to have a fallback mechanism to enable the user to conveniently regain the ability to authenticate if the token is lost or the credentials are forgotten. Other aspects are related to latency in the procedure to regain access to the user account, assurance that recovery of user account is possible for example via built-in backups or secondary recovery schemes.

4.10.2 Security

According to (J. BONNEAU et al., 2012), Any web authentication technique's security is measured using a qualitative approach that evaluates it against 11 specialized security principles:

1. **Resilient-to-Physical-Observation:** an attacker could not break the authentication scheme security by observing the user authenticating one or more times; such observation attacks include shoulder surfing, filming the keyboard, recording keystroke sounds, or thermal imaging of keypad. A

scheme is rated as **Quasi-Resilient-to-Physical-Observation** if it could be broken only by repeating the observation more than 10–20 times.

2. **Resilient-to-Targeted-Impersonation:** the authentication technique is immune to attacks from attackers that exploit the user's personal information (birth date, marital status, family information) to impersonate the user. Personal knowledge questions are the canonical scheme that fails on this point.
3. **Resilient-to-Throttled-Guessing:** the authentication scheme needs to constrain or throttle the attacker's guess rate by implementing policies in the web server to limit the number of password guess attempts; for example, to 5 times per account per day. Any scheme that do not offer this benefit will get a penalty using this evaluation framework.
4. **Resilient-to-Unthrottled-Guessing:** the space of authentication credentials should be large enough such that brute force attacks could not compromise more than 1% of the users accounts. This benefit could be granted to an authentication scheme if an attacker who is capable of attempting up to 2^{40} or even 2^{64} guesses per account could still only compromise less than 1% of accounts.
5. **Resilient-to-Internal-Observation:** the authentication scheme is immune to attacks that intercept the user's input on his machine (using key loggers) or eavesdrops clear text communication (or TLS based) to impersonate the user account. An authentication scheme is granted **Quasi-Resilient-to-Internal-Observation** if the scheme could be broken only by intercepting input or eavesdropping clear text more than 10–20 times.

6. **Resilient-to-Leaks-from-Other-Verifiers:** Nothing that a verifier could possibly leak can help an attacker impersonate the user to another verifier. This penalizes schemes where insider fraud at one provider, or a successful attack on one back-end, endangers the user's accounts at other sites.
7. **Resilient-to-Phishing:** an attacker who compromises users accounts -by fooling them to trust him as a valid verifier to their intended websites- cannot use those credentials to impersonate them against the real website. This penalizes schemes allowing phishers to get victims to authenticate to lookalike sites and later use the harvested credentials against the genuine sites.
8. **Resilient-to-Theft:** if the authentication scheme uses a physical object to complete the user's authentication or part of the authentication process, an attacker who gains or possess this object should not be able to impersonate the user against the website. The scheme is rated **Quasi-Resilient-to-Theft** if the protection is achieved with the modest strength of a PIN, even if attempts are not rate controlled, because the attack does not easily scale to many victims.
9. **No-Trusted-Third-Party:** if the authentication scheme uses a trusted third party as part of the user authentication process, any compromise of the trusted third part that renders it untrustworthy should not make the user account vulnerable to security or privacy threats.
10. **Requiring-Explicit-Consent:** the authentication process must not be started without the explicit consent and initiation from the user.

11. **Unlinkable:** for authentication schemes that are implemented by colluding verifiers, those verifiers cannot determine, from the authenticator alone, whether the same user is authenticating to both.

4.10.3 Deployability

According to (J. BONNEAU et al., 2012), Any web authentication technique's **deployability** features and benefits should be evaluated against the following deployability principles:

1. **Accessible:** the authentication should be accessible; such that users who are able to use passwords for authentication should be able use the scheme for authenticating themselves against a website regardless of any disabilities or any other physical conditions.
2. **Negligible-Cost-per-User:** the cost of the authentication scheme for the user, prover and verifier should be negligible.
3. **Server-Compatible:** a web application provider should not change its existing authentication setup to support the authentication scheme.
4. **Browser-Compatible:** users are not required to change their client to use the authentication scheme, and it is expected to work in cross-browser support. Any browser that is up-to-date and supports JavaScript and HTML5 is expected to be compatible without the need to install any plugins or to perform any configurations. The authentication scheme is rated as Quasi- Browser-Compatible if it needs non-standard but very common plugins, e.g., Flash and java applets.
5. **Mature:** the authentication scheme is already implemented and deployed in production environments and already adopted by large scale of vendors.

6. **Non-Proprietary:** any one can implement or use the authentication scheme publicly without having to pay royalties to anyone. This implies that the authentication scheme is open, well documented and enough resources are publicly available to help users to configure and use it.

4.11 EARMAT evaluation analysis

Table 4 depicts the evaluation results of EARMAT compared to another four popular authentication schemes; the evaluation process is based on the 25 metrics of usability, deployability and security benefits presented in the previous sections.

In terms of Usability, EARMAT, like others, assumes that the user memorizes his password, so it is not Memorywise-Effortless. In regards to Scalable-for-Users property, the current implementation of the protocol assumes that each web application needs its own mobile application to be installed on the client mobile to complete the authentication process; we rate the protocol as Somewhat scalable for users based on that it is easy to manage more than one application in the user's mobile, taking into account that the user's intervention in the authentication process is limited to responding to the authentication confirmation request only. It is theoretically and practically feasible to alter our model such that only one mobile application is used to manage the user's authentications on more than one web application. And this will be our future work.

EARMAT achieves Somewhat Nothing-to-Carry property, and fully achieves the Quasi-nothing-to-carry usability feature, as mobile phones are ubiquitous these days.

Table 4: Comparison between EARMAT, PASSWORDS, GOOGLE 2-STEP VERIFICATION (2SV), PHONEAUTH (IN STRICT MODE) and CAMAUTH.

Y=offers the benefit, S=somewhat offers the benefit.		Passwords	Google 2SV	PhoneAuth	CamAuth	EARMAT
Usability Features	Memorywise Effortless					
	Scalable for Users	Y				S
	Nothing to Carry	Y	Y	S	S	S
	Quasi Nothing to-Carry	Y	Y	Y	Y	Y
	Easy to Learn	Y	S	Y	S	Y
	Efficient to Use	S	S	S	S	Y
	Infrequent Errors	Y	S	S	S	Y
	Easy Recovery from Loss	Y	S	S	S	S
Deployability Features	Accessible	Y	S	Y	S	Y
	Negligible Cost Per User	Y		S	S	Y
	Server Compatible	Y		S	S	Y
	Browser Compatible	Y	Y	S	S	Y
	Mature	Y	Y			
	Non Proprietary	Y		Y	Y	Y
Security Features	Resilient to Physical Observation			Y	Y	Y
	Resilient to Targeted Impersonation	S	S	Y	Y	Y
	Resilient to Throttled Guessing		Y	Y	Y	Y
	Resilient to Unthrottled Guessing			Y	Y	Y
	Resilient to Internal Observation			S	S	Y
	Resilient to Leaks from Other Verifiers		Y	Y	Y	Y
	Resilient to Phishing		Y	Y	Y	Y
	Resilient to Theft	Y	Y	Y	Y	Y
	No Trusted Third Party	Y	Y	Y	Y	Y
	Requiring Explicit Consent	Y	Y	Y	Y	Y
	Unlinkable	Y	Y	S	Y	Y

The properties of Easy-to-Learn, Efficient-to-Use and Infrequent-Errors are also achieved as mobile applications in general are very common nowadays. Easy-Recovery-from-Loss is Somewhat offered, like other authentication schemes, EARMAT will work with SMS-based or email-based OTP in case of failure to use the second authentication factor.

In terms of Deployability features, EARMAT is superior to Google 2SV, PhoneAuth and CamAuth schemes in achieving the Accessible, Negligible-Cost-for-Users, Server-Compatible, and Browser-Compatible features. Our protocol introduces zero configurations or changes to the user's browser, web server O.S. or mobile O.S. In addition to adding zero cost for either the website vendor or the user. For the Mature property, we think EARMAT is able to be a mature authentication scheme, but as this requirement is measured after putting the protocol in production environment; we cannot in this phase empirically verify the mature property. The Non-Proprietary feature of EARMAT is achieved; no proprietary software, hardware or service is necessary for the protocol to work successfully; for GCM, we use it as a communication medium only, and it is implemented as one of existing set of free alternatives including Pushy service (PUSHY).

In terms of Security features, EARMAT is resilient to physical Observation, Targeted Impersonation, Throttled and Unthrottled Guessing, because the attacker will not be able to access the user's account even if he possesses his password until he gains access to his mobile device. Also, attacking the generated authentication token (including the OTP) will not enable the attacker to gain access to the user's account because the OTP is session-specific and not valid for any other web session. The protocol is also resilient to Internal Observation and to leaks from other verifiers, this resilience is achieved due to the session-specific OTP, data communication is carried over secure channels and Private

keys are stored in protected areas in the mobile phone (either hardware protected areas (if supported by the mobile) or system key store).

EARMAT is certainly resilient to phishing attacks and Theft; because compromising the user's credentials by an attacker is not enough to impersonate the user; any authentication attempt needs to be confirmed by the user using his mobile app before being successful. In fact, EARMAT not only neutralizes phishing attacks, but also notifies the user instantly of any suspected illegal attempt to access his account.

EARMAT achieves the No-Trusted-Third-Party feature, as its dependence on GCM is for convenience purposes; i.e. showing up a notification to the user that an authentication process on his account is taking place. While this feature is very important for online notification of possible authentication attacks, it is possible to deactivate this feature and rely on the user himself to start the mobile authentication steps.

EARMAT clearly achieves the Requiring-Explicit-Consent and Unlinkable feature, as the authentication is completed after user confirmation; i.e. no authentication process can be completed on the user's account without his explicit acceptance.

4.12 Summary and conclusions

This chapter introduced a new authentication technique for web-based application that is capable of combating phishing attacks rather than relying on user's experience or phishing detection mechanisms. EARMAT exploits the user's mobile phone to host and operate a mobile app that completes the authentication process as a second level augmenting the traditional password-based web authentication.

EARMAT authentication strength point is in its ability to instantly notify the user about any login attempt on his account (using GCM), and powering the user to confirm or reject that login attempt, which makes it –by nature- immune to phishing

attacks; as compromising the user credentials alone is not enough to compromise the user's account because the authentication is verified using the user's mobile app as a second level.

EARMAT implements RSA cryptography and digital signatures to mutually authenticate both the web site client and the server, which adds a new level of security that prevents Man in the Middle Attacks.

A system prototype of the authentication protocol was implemented and analyzed to ensure its feasibility. In addition, we evaluated the protocol against the 25 features of the assessment framework regarding Usability, Deployability and Security of web authentication schemes. In future, we plan to extend EARMAT to support single mobile application to manage a set of user accounts to further improve the protocol scalability of users' property.

5 A Bastion MobileID-Based Authentication Technique

Despite their proven security breaches, text passwords have been dominating all other methods of human authentication over the web for tens of years, however, the frequent successful attacks that exploit the passwords vulnerable model raises the need to enhance web authentication security. In this chapter, we will discuss BMBAT; a new proposed authentication technique to replace passwords, that leverages the pervasive user mobile devices, QR codes and the strength of symmetric and asymmetric cryptography. In BMBAT, the user's mobile device acts as a user identity prover and a verifier for the server; it employs a challenge-response model with a dual mode of encryption using Advanced Encryption Standard (AES) and Public Key Infrastructure (PKI) to mutually authenticate the client to the server and vice-versa. BMBAT contributes to the strategy of combating phishing attacks by strengthening the website authentication process by introducing the password-less authentication scheme and thus making the authentication process tightly coupled with the user's mobile device. BMBAT combats a set of attack vectors including phishing attacks, man in the middle attacks, eavesdropping and session hijacking. A prototype of BMBAT has been developed and evaluated; the evaluation results show that BMBAT is a feasible and competitive alternative to passwords.

5.1 Password based authentication overview

The rapid and continuous advances in web technologies have made the internet industry to be part of almost everyone's today life; from the widely spread social media to companies and financial institutions that offer their services online; thus imposing the very importance of securing the world wide web, especially for end users' private data. One aspect of a website security is the process of authenticating end users to the website services, while the traditional text-passwords are the dominant option for end user authentication, huge efforts -in both the industrial and academic sectors- have been

conducted to replace this sticky password scheme; researches concluded that passwords are vulnerable to security attacks (J. BONNEAU et al., 2012) including brute-force attacks, guessing, key- loggers, phishing attacks, malwares, eavesdropping, dictionary attacks, etc.

Password authentication have been analyzed over years in an attempt to identify its potential strength and weakness points, a survey of corporate users (A. ADAMS AND M. SASSE, 1999) found that users are confused by the password policy requirements and that they write their passwords on posting notes, thus compromising the security of their accounts. A study on (S. GAW AND E. W. FELTEN, 2006) found that users are registered in accounts for which they forgot their passwords and even confused of whether they registered in such accounts or not, so users are usually overwhelmed by the number of passwords they maintain for different websites.

According to (J. BONNEAU et al., 2012), three factors contribute to the success of any web authentication scheme: usability, deployability and security. In terms of usability, passwords are considered good in achieving usability measures, as they are easy to learn, efficient to use (just typing a few letters) and easy to recover from loss as websites usually provide fallback mechanisms for users to recover their passwords.

Passwords most success is in their deployability features; they are accessible, incorporate zero cost for both users and vendors, compatible with both server and browser infrastructure, mature and not proprietary.

Regarding security, passwords evaluated to be poor in security measures; they are not resilient to physical observation as they can be automatically observed by key loggers or through a high quality video recording of the keyboard (D. BALZAROTTI et al., 2008).

Also, passwords are vulnerable to dictionary attacks and guessing attacks, not resilient to leaks from other verifiers and most importantly, they are not resilient to phishing attacks – our concern-.

The poor security evaluation of passwords made users and website vendors to accept to compromise their usability and deployability features for the benefit of more secure authentication schemes. In this work, we propose and implement an authentication scheme that replaces password authentication, giving the required security level for users and their private data while keeping an accepted level of usability and deployability features, this balance makes the proposed scheme feasible and possible to be implemented and adopted by websites.

5.2 Proposed Authentication Scheme

BMBAT is designed to meet a set of design guidelines; including achieving mutual authentication between the server and the user through his mobile device, supporting multiple user accounts on different web applications and a secured communication channel (using SSL) between the user mobile and the server.

The proposed authentication method applies the concept of mutual authentication so that the web server proves its identity to the user mobile client and the user mobile client proves its identity to the server before authenticating the user session. As depicted in figure 33, the proposed mutual authentication process incorporates five steps:

Step1: the user initiates the process by presenting his credentials through the web browser to the web server.

Step2: the web server validates the user credentials and prepares the authentication token using the server certificate private key and the shared key, and then encapsulates it with a QR code and send it back to the user browser.

Step3: The user scans the QR code with his mobile app, processes the login ticket by decrypting it using the server's public key and the shared key to verify the server's identity, and confirms the user identity to the server or extracts the login code and displays it to the user.

Step4: The application server receives the decrypted nonce, verifies the user identity and authenticates the user's session.

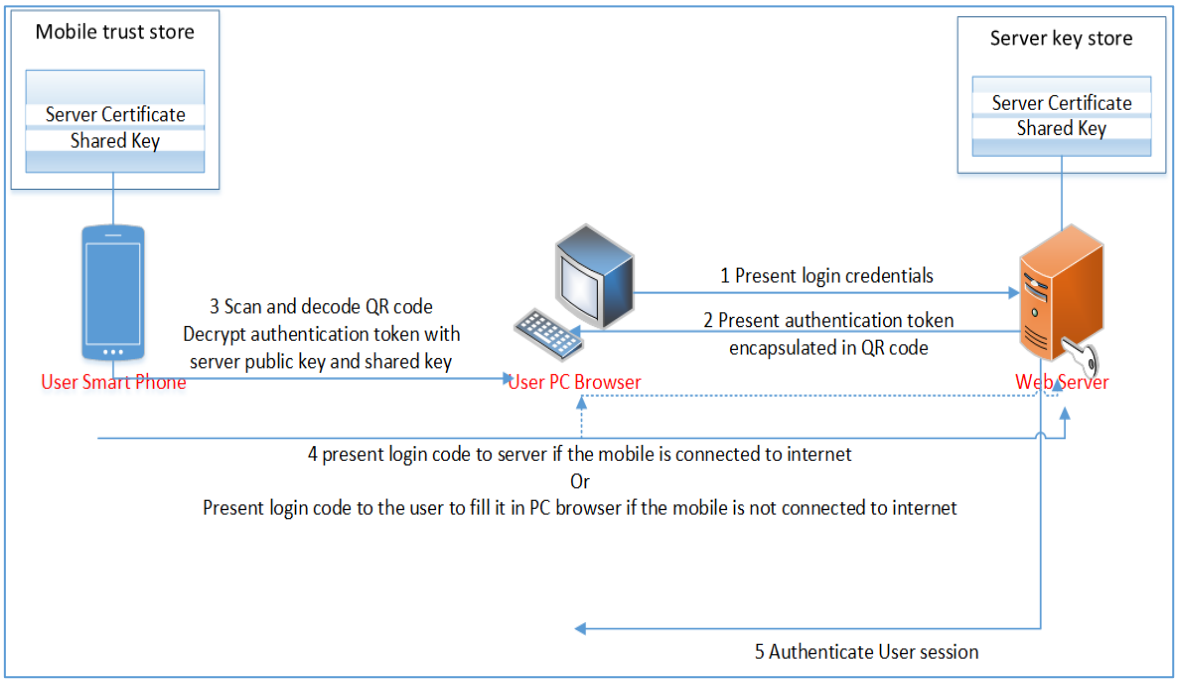

Figure 33: Proposed Mutual Authentication Process

5.3 BMBAT Account Creation

In order for a user to be registered in a website the applies BMBAT for users' authentication, the user needs to complete a set of steps to identify himself to the website; Figure 34 depicts the main scenario of the registration process. Table 5 summarizes the components notations used in BMBAT.

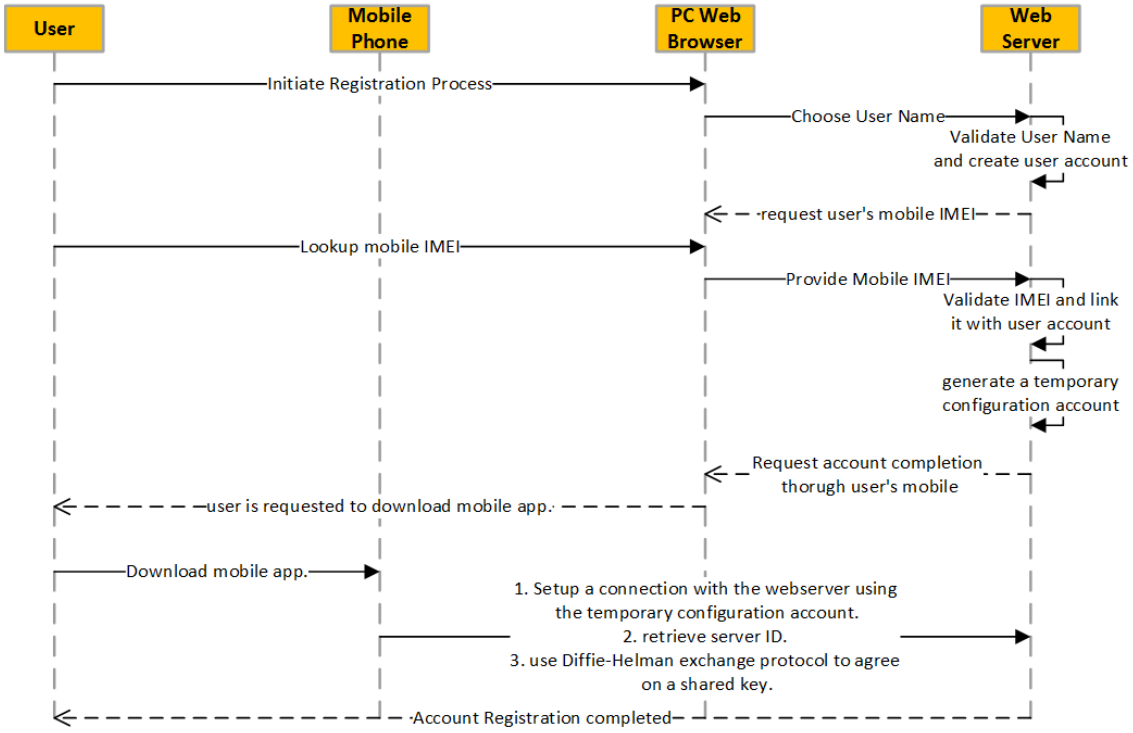

Figure 34: BMBAT Registration Steps

Table 5: Table of notations

Symbol	Name	Description
N	Nonce	Randomly generated 10 alphanumeric characters.
MTS	Mobile Time Stamp	Used to specify a validity time for mobile response.
STS	Server Time Stamp	Used to specify a validity time for the login code
SID	Session Identifier	A web server auto generated string that identifies each web session.
Lcode	Login Code	The challenge that is encoded in the QR code
K	Shared key	256 bit AES key that is shared with the web server for each user.
P_r	Server Private Key	The web server private key (RSA 2048 bit), accessible only to the web server.
P_u	Server Public Key	The web server public key, shared among all users and is publicly available.
Λ	Threshold Value	The validity period (in minutes) of the login code.
ServerID	Server ID	Web application ID to be associated with the user name in BMBAT.
User_ID	User Name	User Name identifier.

Step1: the user is required to specify a valid user name that identifies his account and submit it to the web server along with his mobile IMEI (International Mobile Equipment Identity) code and a valid email address to initiate the account creation process.

Step2: the web application validates the user name, user email and the IMEI code, and then creates the user account and generates a temporary configuration account credentials to be used by the user to complete the registration process from his mobile device.

Step3: the user needs to install the BMBAT app in his mobile device if it is not already there.

Step4: the user uses the temporary configuration account that was generated in step2 to connect to the web app through BMBAT app, at this stage BMBAT completes the user registration by retrieving the server's public key and the server ID and associating it with the user name and then establishes an agreement with the web server on the shared key; using Diffie-Hellman key exchange protocol.

The communication between BMBAT and the web application at the registration phase is assumed to be carried over a secured connection (typically HTTPS) to protect the registration data from any unauthorized access.

The association of user name with a server ID is necessary for BMBAT to support multiple accounts on different web applications for the same user. This association is saved in a secure manner in the user's mobile. The shared keys are also stored in a secure trust store in the mobile phone, we think mobile devices trust stores are secure enough to protect the shared key from unauthorized access, thanks to TrustZone technology (TRUSTZONE) that is implemented in most modern mobile devices.

5.4 BMBAT User Login

The login phase consists of a set of steps as depicted in Figure 35, and discussed as follows:

5.4.1 Login initiation

when the user wishes to login to the website, he requests the website login page through his “untrusted” browser, then the web server responds with a login page requesting him to fill his user name in addition to filling the Captcha

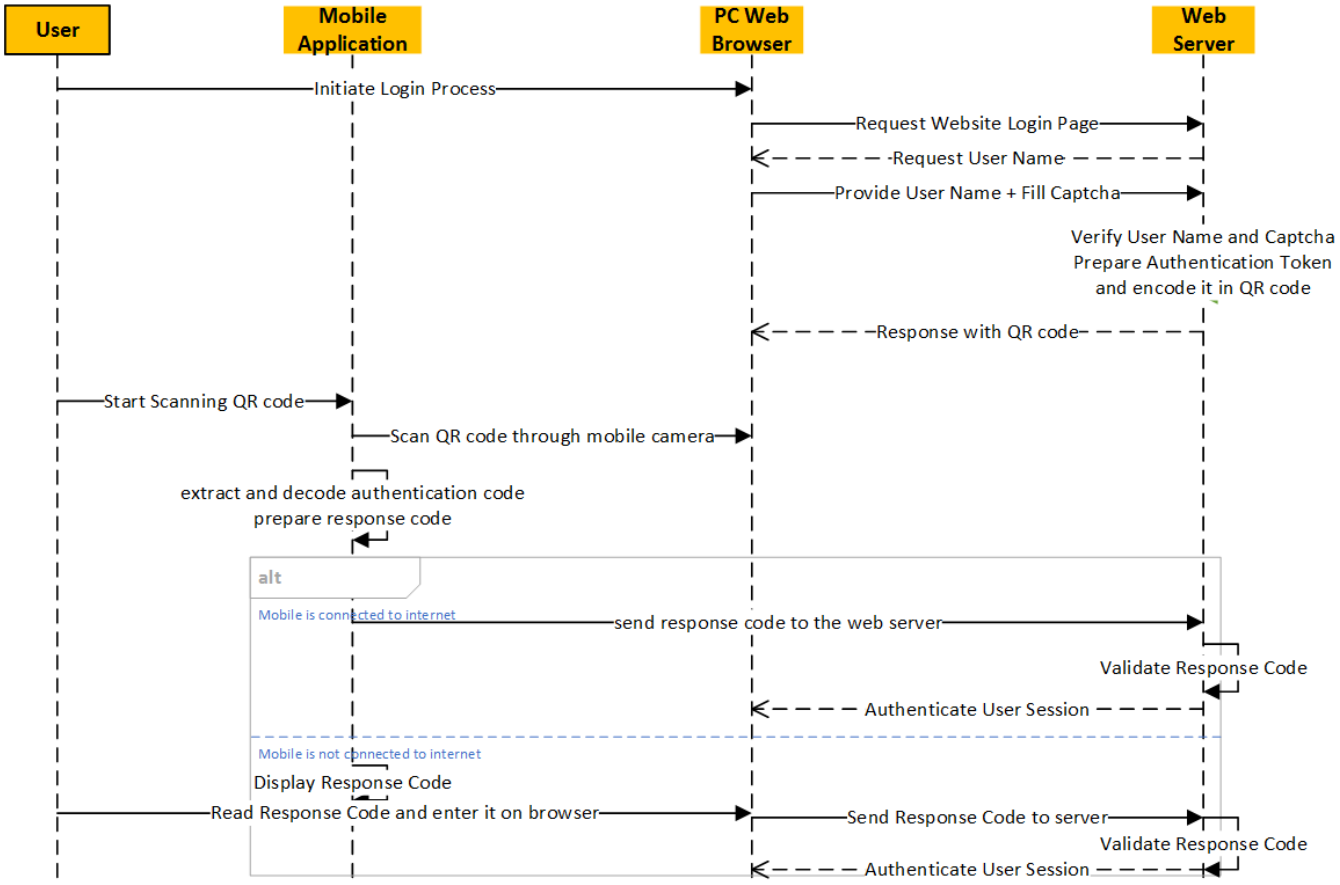

Figure 35: Login Phase Sequence Diagram

component; the requirement of filling the captcha component is added to prevent attackers from overwhelming the server with auto-generated login requests; built on the assumption that the user name could be publicly known.

5.4.2 User name verification

once the server receives the user name, it verifies that it refers to a registered user, upon success, the server initiates the task of creating the user authentication token.

5.4.3 User Authentication Token

In response to the user login request, the web server prepares a Login Code for that user's session; it generates a random session nonce (N) and associates it with the user session (SID), server current timestamp (STS) and user name, then it encrypts it with the shared key (K), the resulting cipher is then concatenated with the user name and Server Authentication Service API and then digitally signed with the server's private key (P_r). the resulting authentication token and ServerID are encoded in a QR code and sent back to the user's browser. An XML representation of the authentication token contents could be as follows:

{

ServerID: *5T208*

Token: *"15g8T45Krco…"*

Server_Auth_API: *"https://login.examplesite.com"*

}.

The login code generation algorithm is depicted in figure 38.

5.4.4 Response

The encoded QR code is sent back as a response to the user's browser, the response page also offers an option for the user to enter the deciphered login code as will be explained next.

5.4.5 QR code scanning

Now the user needs to scan the QR code that is displayed on the PC browser to complete authenticating himself to the web site, this scan is done using the user's mobile application that was associated with his account in the registration process.

Once the authentication token is extracted from the QR code in the mobile application, the following actions and options take place (see figure 37):

- The authentication token is decrypted using the server's public key, to extract the user name and the encrypted nonce.
- BMBAT looks up the shared key based on the user name and the ServerID.
- The encrypted nonce is then decrypted using the pre-shared key to extract the login token and the associated time stamp (STS) and session identifier SID.
- The login token is now available for BMBAT, and its validity is determined based on the time stamp of the token creation time and a threshold value (λ).
- The next action is determined based on whether the mobile is connected to the internet or not; If the mobile has access to the internet, the login token is augmented with the mobile timestamp and SID and sent back to the server directly (Server_Auth_API) over a secure connection. The server verifies the received login token against the user's session and automatically authenticates the user session if the verification succeeds (see figure 36). In case the mobile is not connected to the internet, the mobile application extracts the nonce N from the login code and displays it to the user. Then the user is required to enter this nonce manually in his browser. The server verifies the received nonce against the user's session and authenticates the user session if the verification succeeds.

```
//this algorithm generates the authentication token
Begin
User_id=getParameter(User_ID);
SID=getSessionIdentifier();
N=generateRandomNonce();
STS=getServerTimeStamp();
K=lookupSharedKey(User_ID);
Token=concatenate(N,SID,User_ID,STS);
Token=encrypt(K,Token);
Pr=lookupServerPrivateKey();
encryptedToken=encrypt (Pr, Token);
response=concatenate(ServerID,Server_Auth_API,
encryptedToken);
saveAuthenticationInfo(SID,N,STS,User_ID);
sendResponseQRToClient(response);
End;
```

Figure 38: Token Generation Algorithm

```
// this algorithm applies the server steps to validate
//the client authentication
Begin
Token=getParameter(Token);
User_ID=extractUserID(Token);
SID=extractSID(Token);
MTS=extractMTS(Token);
encryptedNonce=extractEncryptedNonce(Token);
if validSID(SID,User_ID) then
  K=lookupSharedKey(User_ID);
  N=decrypt(K, encryptedNonce);
  If N==loadSavedNonce() and validMTS(MTS,λ)
then
    Authenticate_user_session();
  Else
  Deny_user_session();
  End if;
end if;
End;
```

Figure 36: Server authentication algorithm

```
//this algorithm prepares the authentication token using the
//server's private key and the shared key to be used in the
//mutual authentication process
Begin
authToken=scanQRCode();
ServerID=extractServerID(authToken);
Token=extractToken(authToken);
Pu=lookupServerPublicKey(ServerID);
decryptedToken=decrypt (Pu, Token);
Server_Auth_API= extractServerAuthApi(authToken);
User_ID=extractUserID(decryptedToken);
encryptedNonce=extractNonce(decryptedToken);
K=lookupSharedKey(User_ID,ServerID);
decryptedToken=decrypt(K, encryptedNonce);
SID=extractSID(decryptedToken);
N=extract(decryptedToken);
STS=extractSTS(decryptedToken);
If validSTS(STS,λ) then
  If mobile_connected_internet then
    Response=encrypt(K,N);
    SendResponse(Response+User_ID+SID+MTS,
Server_Auth_API);
  Else
    displayToMobileScreen(N);
  End if;
End if;
End;
```

Figure 37: QR code processing algorithm

5.5 Fall-back Mechanism

In case the user mobile is no longer available (due to theft, damage, etc..), the user needs to disassociate his account from the not accessible mobile device, BMBAT offers the option for the user to deactivate BMBAT authentication by visiting all registered websites

and requesting sending a deactivation email to his email address, which includes a link through which he can remove his mobile device from the trusted devices.

Also another mitigating factors could be used to protect the user accounts in case the mobile is lost; it is possible to damage the mobile phone data using the privileges granted by the phone vendor, also the user is encouraged to configure his mobile so that a pass code (or maybe a finger print) is needed before opening BMBAT app, in this case the attacker needs to bypass the passcode before being able to impersonate the user.

5.6 BMBAT Evaluation and Security Analysis

In order to evaluate BMBAT usability, deployability and security features, we benchmarked it to another four known authentication schemes; based on the web authentication assessment framework proposed by (J. BONNEAU et al., 2012). A set of 25 measures have been applied to BMBAT and compared to other four popular authentication schemes; Passwords, Google 2 step-verification, PhoneAuth (A. CZESKIS et al., 2012) and CamAuth (MENGJUN XIE et al., 2015), the comparison results are shown in Table 6.

The comparative evaluation results show that BMBAT is highly competitive to existing authentication techniques. In terms of usability, BMBAT offers the benefit of eliminating the need for users to remember their passwords and neutralizes password entry errors by eliminating the use of passwords at all, while offering somewhat the benefit of “Nothing to Carry”; as we consider the mobile device to be always side by side to the user.

In terms of deployability, BMBAT is an accessible and zero-cost for the user, but we could not assume at a mature authentication technique as this property needs to be test thoroughly in production environments. The security features of BMBAT enable it to

Table 6: Comparison of BMBAT, Passwords, Google 2-Step Verification (2SV), PhoneAuth (in strict mode) and CamAuth. Y=offers the benefit, S=somewhat offers the benefit.

		Passwords	Google 2SV	PhoneAuth	CamAuth	BMBAT
Usability Features	Memorywise Effortless					Y
	Scalable for Users	Y				S
	Nothing to Carry	Y	Y	S	S	S
	Quasi Nothing to-Carry	Y	Y	Y	Y	Y
	Easy to Learn	Y	S	Y	S	Y
	Efficient to Use	S	S	S	S	Y
	Infrequent Errors	Y	S	S	S	Y
	Easy Recovery from Loss	Y	S	S	S	S
Deployability Features	Accessible	Y	S	Y	S	Y
	Negligible Cost Per User	Y		S	S	Y
	Server Compatible	Y		S	S	Y
	Browser Compatible	Y	Y	S	S	Y
	Mature	Y	Y			S
	Non Proprietary	Y		Y	Y	Y
Security Features	Resilient to Physical Observation			Y	Y	Y
	Resilient to Targeted Impersonation	S	S	Y	Y	Y
	Resilient to Throttled Guessing		Y	Y	Y	Y
	Resilient to Unthrottled Guessing			Y	Y	Y
	Resilient to Internal Observation			S	S	Y
	Resilient to Leaks from Other Verifiers		Y	Y	Y	Y
	Resilient to Phishing		Y	Y	Y	Y
	Resilient to Theft	Y	Y	Y	Y	Y
	No Trusted Third Party	Y	Y	Y	Y	Y
	Requiring Explicit Consent	Y	Y	Y	Y	Y
	Unlinkable	Y	Y	S	Y	Y

highly compete with similar authentication techniques, the offered security features are discussed as follows:

1. Resilient to Physical Observation: BMBAT achieves this property because it does not rely on password authentication, so any physical observation of the authentication process will never reveal any clue on the authentication details. In case the user needs to enter the login code in his browser, the observation of this code is useless as it is a session-based onetime code.

2. Resilient to Targeted Impersonation: an attacker who possess a knowledge of the user's personal details (such as birthdate, relative names etc.) will be not able to impersonate the user, as none of such information is needed or related to any step in BMBAT authentication.
3. Resilient to Throttled and Unthrottled Guessing: an attacker's chance to succeed to guess or brute-force the shared key between the user and the server is very low, and trying to brute-force the authentication nonce is not useful for an attacker as it is valid only for the current user session.
4. Resilient to Internal Observation: An attacker cannot impersonate a user by intercepting the user's input from inside the user's device (e.g., by keylogging malware) or eavesdropping on the clear text messages communicated between prover and verifier; BMBAT achieves this property by using a session-based onetime token that renders its usage again of no benefit, in addition all data communication is assumed to be carried over a secure channel.
5. Resilient to Leaks from Other Verifiers: BMBAT authentication parameters (shared keys) are dedicated for users in a specific web site, so that a successful dictionary

attack on a web site or the user mobile (and thus compromising shared keys) will never make it possible to compromise user accounts in other different websites.

6. Resilient to Phishing: BMBAT neutralizes phishing attacks by eliminating their core attack principle; i.e. no password to be stolen; The phone sends the shared secret, and will only send it to the web site associated with the user name and ServerID that were extracted from encrypted QR code login ticket.
7. Resilient to Theft: we rated BMBAT to be resilient to theft as there are a set of solutions to prevent a stolen user's mobile from being used to access the linked accounts; starting from locking codes that are available on modern mobile phone, to the possibility of deactivating the phone online by service providers. Nevertheless, BMBAT will offer the option for a user to deactivate his account association with the mobile phone in case it was lost; the user can access his account online using a variety of ways including master password or onetime passwords.
8. No Trusted Third Party: BMBAT does not rely on a trusted third party (other than the user mobile and the server) who could, upon being attacked or otherwise becoming untrustworthy, compromise the prover's security or privacy.
9. Requiring Explicit Consent: BMBAT authentication will never be initiated without an explicit action from the user to request authentication; this is achieved through requesting a user name and a Captcha code and then scanning the QR code by the user's mobile.
10. Unlinkable: this property "measures whether Colluding verifiers can determine - from the authenticator alone, whether the same user is authenticating to both" (J. BONNEAU et al., 2012); we rate BMBAT as Unlinkable as the parameters that

identify a user (shared key and user name) need not to have any thing in common when the user is registered to more than one website.

5.7 BMBAT Contributions

BMBAT is designed to offer a better security than text passwords with minimum compromise of usability and deployability for both end users and service providers; it protects against the following attack vectors:

1. Phishing attacks: while traditional phishing attacks succeed to compromise the user's credentials on password-based authentication schemes; the proposed authentication scheme neutralizes them completely; as there is no password for the user to be compromised, the user is identified through his mobile device.
2. Man in the middle attacks: In this class of attacks, the attackers situate themselves between the user and the original web site, and proxy all communications between the user and the real web site, from this point, the attacker can observe and record all transactions including the user's credentials. For such an attacks to succeed, the attacker must be able to direct the user to the attacker's proxy server instead of the real server. This may be accomplished using a number of methods including Transparent Proxies, DNS Cache Poisoning, URL Obfuscation and Browser Proxy Configuration. The proposed scheme combats this type of attacks by:
    - Mutually authentication both parties- the user and server- with the pre-shared key and server's private key respectively.
    - The authentication in the proposed system relies upon a session-based login code; which renders it useless for any attacker to reuse another session's login code.

3. eavesdropping: the proposed system assumes it is possible for an attacker to passively or actively eavesdrop on any network communication between the user and the server in the login phase; as the traffic data is either protected by a dual encryption mode with the pre-shared key and server's private key or the data is session based; i.e. it is valid only for the current session.
4. Dictionary attacks: the proposed system is not immune to dictionary attacks; as the keys shared with the system users need to be maintained in somehow on the server machine. Any compromise of the shared keys would put the user accounts in risk.
5. Session Hijacking: BMBAT assumes that the data communicated between different parties in the authentication process is protected using HTTPS protocol, such that it is not possible for an attacker to exploit a valid session to gain unauthorized access to the website in behalf of the user.

5.8 BMBAT Implementation and performance analysis

We have implemented a prototype system for BMBAT authentication protocol, including a web application and mobile application to act as an identity prover. We built a java-based web application that handles a set of server side services for the authentication protocol including token generation, encryption, QR code generator and an authenticator. the client's mobile application is developed on android 5.1, and its compatible with android 2.2 and upward platforms, the mobile application is responsible for device registration, maintaining user shared keys, confirming and completing the authentication process, the application uses the necessary APIs for key exchange, storage, encryption, decryption and communications with the server.

To assess the performance of BMBAT, we conducted a performance test using an emulator in android studio with the following specifications:

- Device: Nexus 5
- CPU: x86
- RAM: 1.5 GB
- Platform Version: Android 5.1 (lollipop)

In this evaluation we measured two metrics of the non-functional requirements of the system:

- Response time: the time in seconds that BMBAT mobile app takes to execute the algorithm of processing the QR code figure 5.
- Memory usage: the volume in megabytes that BMBAT decryption process used in the RAM of the mobile execution environment.

The results of the performance test showed that BMBAT -in average- spent 5.66 Milliseconds in executing the algorithm of processing the login code, while consuming 0.08 MB of memory. The whole memory reserved by the application was 2.57 MB. Table 7 depicts the test results of six runs of the algorithm of decrypting the login code using the emulator.

Table 7: Login code processing algorithm performance on the emulator

Run #	Execution Time (ms)	CPU Usage	Memory Usage (MB)
1	7	15%	0.09
2	4	12%	0.08
3	6	12.5%	0.07
4	5	9.5%	0.06
5	8	16%	0.07
6	4	18%	0.04

The complete authentication process will include also the time spent in scanning the QR code and communicating the login token to the server, the scan process is assumed intuitive and easy operation for most users, and the process of sending back the login token to the server is an automated process that is expected to add a very little time fraction (in milliseconds).

These performance test results indicate that BMBAT implementation in most modern mobile devices will be feasible and the response time will be accepted by users, making it possible to adopt such an authentication scheme at a compromise of a couple of seconds latency.

5.9 Future Work

The emergence of the Internet of Things "IoT" opens the door for smaller and maybe more ubiquitous user devices that could be a better replacement for user's mobile phone for identity proof. Therefore, the future may raise the need to reevaluate the cost of executing cryptographic algorithms (either symmetric or asymmetric) on such devices. One prominent choice for replacing such algorithms in BMBAT is to use the less costly elliptic curve cryptography which requires smaller keys compared to AES or RSA but offers the same security levels, thus needs far less processing than current RSA or symmetric key cryptography.

5.10 Summary and conclusions

In this work, we presented BMBAT, a new web authentication scheme that employs the user's mobile phone as an identity proofer. This technique employs a dual encryption mode (RSA and AES) to implement a challenge –Response mechanism that enables the web server to identify the user identity in an easy and smart manner. The mobile application completes the response phase by either sending the response directly to the server or by displaying the response code to the user in case the mobile is not connected

to the internet. Our evaluation and security analysis of BMBAT concluded that it is a competitive alternative to traditional password-based web authentication and that it overcomes all security breaches that are plausible to passwords. Moreover, we implemented a prototype of BMBAT to prove that it is applicable and a feasible alternative to current web authentication methods.

6 An intelligent classification model for phishing email detection

This chapter presents an intelligent classification model for detecting phishing emails using knowledge discovery, data mining and text processing techniques. This classification model introduces the concept of phishing terms weighting which evaluates the weight of phishing terms in each email. The pre-processing phase is enhanced by applying text stemming and WordNet (GEORGE A. MILLER, 1995) ontology to enrich the model with word synonyms. The proposed model applied the knowledge discovery procedures using five popular classification algorithms and achieved a notable enhancement in classification accuracy; 99.1% accuracy was achieved using the Random Forest algorithm and 98.4% using J48, which is –to our knowledge- the highest accuracy rate for an accredited data set. A comparative study with similar proposed classification techniques is also introduced to evaluate the proposed classification model.

Our focus in this chapter is to build an intelligent classifier at the email level that is capable of detecting phishing emails as an early stage in the phishing combating process. We believe that detecting phishing emails can make the internet users more secure by eliminating those emails and not relying on the users' vigilance to protect their data from phishing attacks. Many studies concluded that depending on human factors is not a preferred option for combating phishing attacks; especially for advanced and well prepared phishing attacks that are continuously adapting themselves to known defense mechanisms (RACHNA DHAMIJA et al., 2006) (JULIE S. DOWNS et al., 2006).

Our approach for detecting phishing emails applies the knowledge discovery model and data mining techniques to build an intelligent model that learns from existing training dataset of both ham and phishing emails, the model extracts and reduces the important features that contribute to building a set of classifiers from which the best classifier is

chosen. The proposed model builds a java program that extracts a set of features from the email header and body; those features are then augmented with a weighted term frequency that is applied after performing linguistic processing of the email extracted terms. After that a set of data mining algorithms are applied to the extracted features to decide the algorithm with best results.

6.1 Proposed Model

The proposed approach for phishing email classification employs the model of Knowledge Discovery (KD) and data mining for building an intelligent email classifier that is able to classify a new email message as a legitimate or spam; the proposed model is built by applying the iterative steps of KD to identify and extract useful features from a training emails data set, the features are then fed to a group of data mining algorithms to identify the best classifier.

The proposed model for email classification utilizes linguistic processing techniques and ontologies to enhance the similarity between emails with similar semantic term meanings. Also the principle of term document frequency is applied in weighting the phishing terms in each email such that emails phishing terms weighting helps in discriminating phishing from legitimate emails.

The proposed model reduced the number of features used in the classification process into 16 features only; which enhances the classification performance and efficiency and minimizes the noise of including many features and hence improves the classification accuracy. These enhancements are discussed in detail in the following subsections.

6.2 Knowledge Discovery Model

Knowledge discovery is the process of extracting or discovering patterns from data, the extracted patterns should be novel (not known in prior), valid (generalized to new data),

useful (lead to useful actions or decisions) and understandable (lead to human understanding of the data). The KD process is carried out using a set of iterative steps as depicted in figure 39. The steps are initiated by understanding the problem and the data, followed by a data pre-processing phase to prepare it for the data-mining step through which the target knowledge is discovered, evaluated and then presented as a useful and easy to use information.

The set of KD steps are briefly explained as follows:

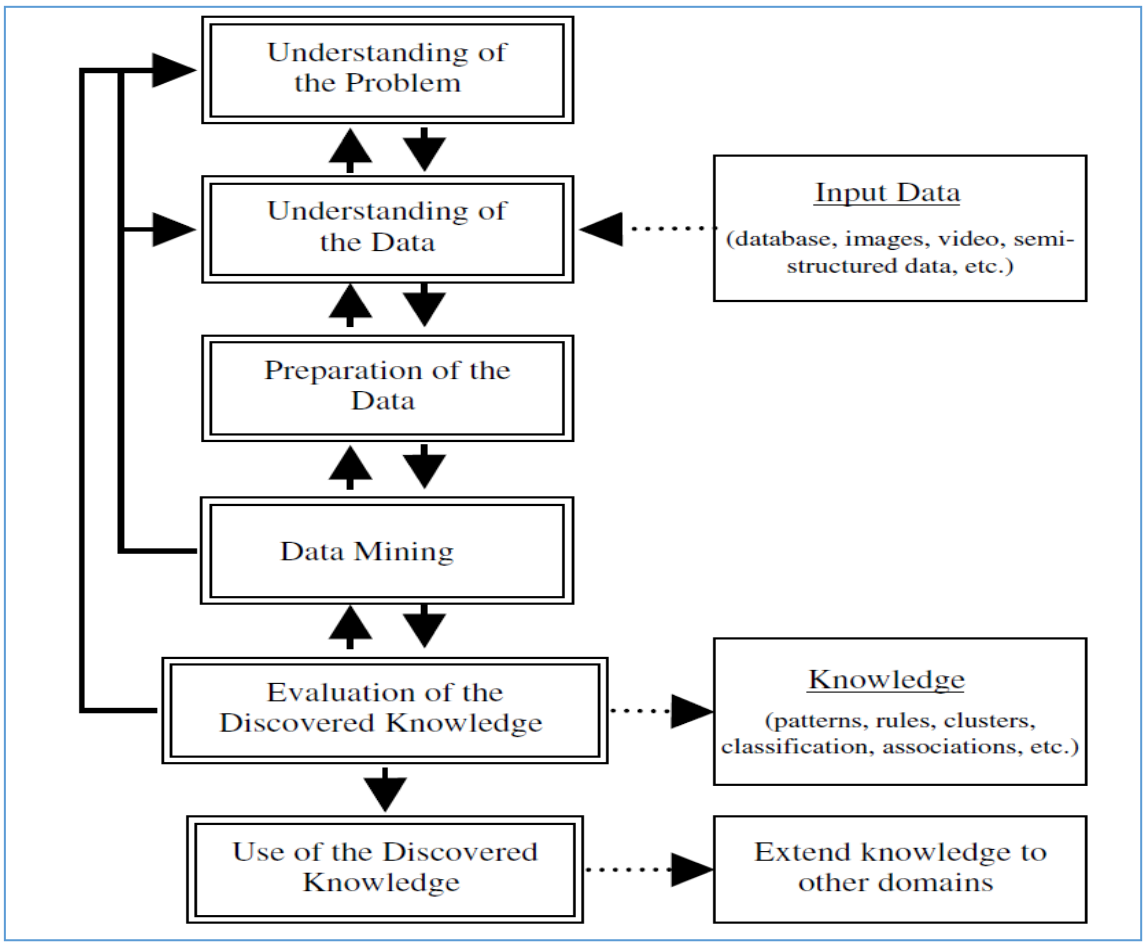

Figure 39: Knowledge Discovery Process (PAL et al., 2005)

- **Understanding the problem domain**: In this step, it is important to work with domain experts to understand the problem, specify goals and understand current solutions of the problem.

- **Understanding the data**: This step includes deciding and collecting sample data about the problem and its format and size, this could be a local empirical data or from an accredited public data set. Usually the data is checked for consistency, redundancy, missing values, etc. and proper pre-processing actions take place to correct the data.
- **Data preparation**: in this step the collected data is cleaned and pre-processed to fix errors, process redundant and missing values, also the set of features are selected and evaluated to decide the best set of features that contribute to the best solution of the problem. The feature selection process is carried out using specialized feature selection and evaluation algorithms and techniques, e.g. the information gain technique.
- **Data Mining**: in this step, the chosen data mining techniques and algorithms are applied on the training data set that was prepared in the previous steps. The process of data mining could involve association pattern extracting, clustering, classification or other techniques according to the problem solution requirements. The output of this step is the discovered knowledge that leads to a decision or helps in taking the needed actions.
- **Knowledge evaluation**: in this step the discovered knowledge from the data mining step is evaluated and its correctness, novelty and usefulness is tested.
- **Discovered knowledge usage**: in this final step the useful discovered knowledge is deployed and put in action as a solution or part of a solution to the problem in the hand.

The proposed model architecture is depicted in figure 40 and explained subsequently.

6.3 Data Collection Phase

The first step in building the proposed phishing email classifier is choosing the suitable training data set which is a real sample of existing emails that consists of both phishing and legitimate emails (also known as spam and ham emails). The training data set will be used to discover potentially predictive relationships that will serve as building blocks in the classifier. Our training data set consists of 10538 emails including 5940 ham emails from spam assassin project (J. MASON, 2005) and 4598 spam emails from Nazario phishing corpus (J. NAZARIO).

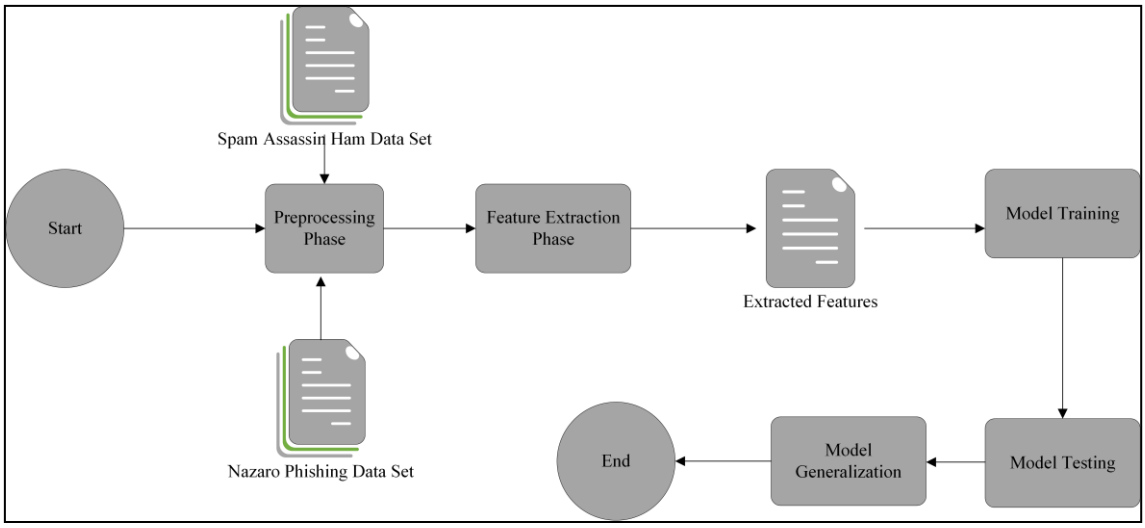

Figure 40: The proposed model architecture

6.4 Data Pre-processing and features extraction

In this step, the emails in the training data set are prepared and filtered such that they can be transformed into a data format that is easily and effectively processed in subsequent steps of building the classifier. The emails in our chosen training data set are available in plain text format which needs to be pre-processed and transformed into EML format (Microsoft Outlook Express file extension) that is interoperable with the java mail package that will be used to extract the email features. Figure 41 depicts the main actions that take place in the pre-processing step.

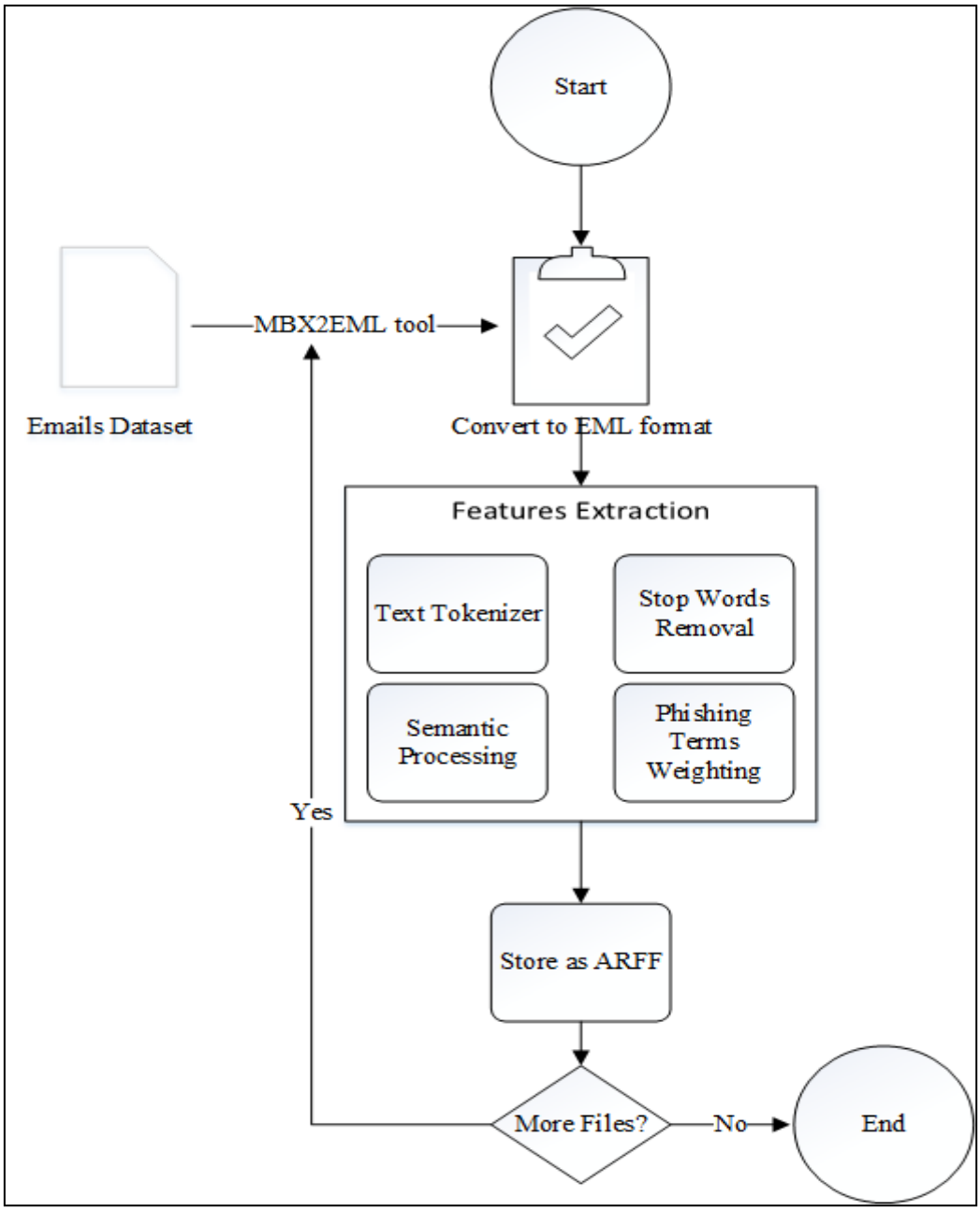

Figure 41: Pre-processing Phase

The proposed mail classification model utilizes a set of 16 extracted features from the email message header and body; the extracted features are explained in table 8.

The process of extracting the features set from each email utilizes a java program that reads each email in the training data set parses its contents and computes the value for each feature according to its description. After extracting the feature set for each email, it

Table 8: Email extracted features

Feature	Description	Data Type	Information Gain
HTML Body	Checks if the email body contains HTML content.	Number {0,1}	0.681
Hexadecimal URLs	The number of URLs consisting of hexadecimal characters in the email.	Number	0.652
Domains Count	The number of domains in the URLs that exists in the email.	Number	0.652
TextLinkDifference	The number of URLs whose label is different from its anchor in the email.	Number	0.649
Dots Count	The maximum number of dots that exist in a URL in the email.	Number	0.497
Email Contains Account Term	Checks if the email contains the term "Account"	Number {0,1}	0.493
Email Contains Dear Term	Checks if the email contains the term "Dear"	Number {0,1}	0.375
Images as URL	The number of image URLs.	Number	0.298
IP URLs	The number of URLs whose domain is specified as an IP address.	Number	0.297
Email Contains PayPal Term	Checks if the email contains the term "PayPal"	Number {0,1}	0.296
Email Contains Login Term	Checks if the email contains the term "Login"	Number {0,1}	0.250
Email Contains Bank Term	Checks if the email contains the term "Bank"	Number {0,1}	0.213
Phishing Terms Weight	A weight that is assigned to each email and represents the sum of weights of the phishing terms that exists in that email	Number	0.210
Email Contains Verify Term	Checks if the email contains the term "Verify"	Number {0,1}	0.207
Email Contains Agree Term	Checks if the email contains the term "Agree"	Number {0,1}	0.206
Email Contains Suspend Term	Checks if the email contains the term "Suspend"	Number {0,1}	0.205

is written into an ARFF (Attribute-Relation File Format) file that will be fed later into the classifier building process.

We used the Information Gain (IG) measure to specify the usefulness of each feature in our features set in discriminating between the spam and ham classes, the IG value for each attribute tells us how important a given feature of the feature vectors is. The IG for each feature is depicted in table 8. We found that the features listed in table 8 has the highest IG value which indicates that they will have an important contribution in deciding the email class as phishing or legitimate. Figure 42 depicts the IG values of our proposed feature set.

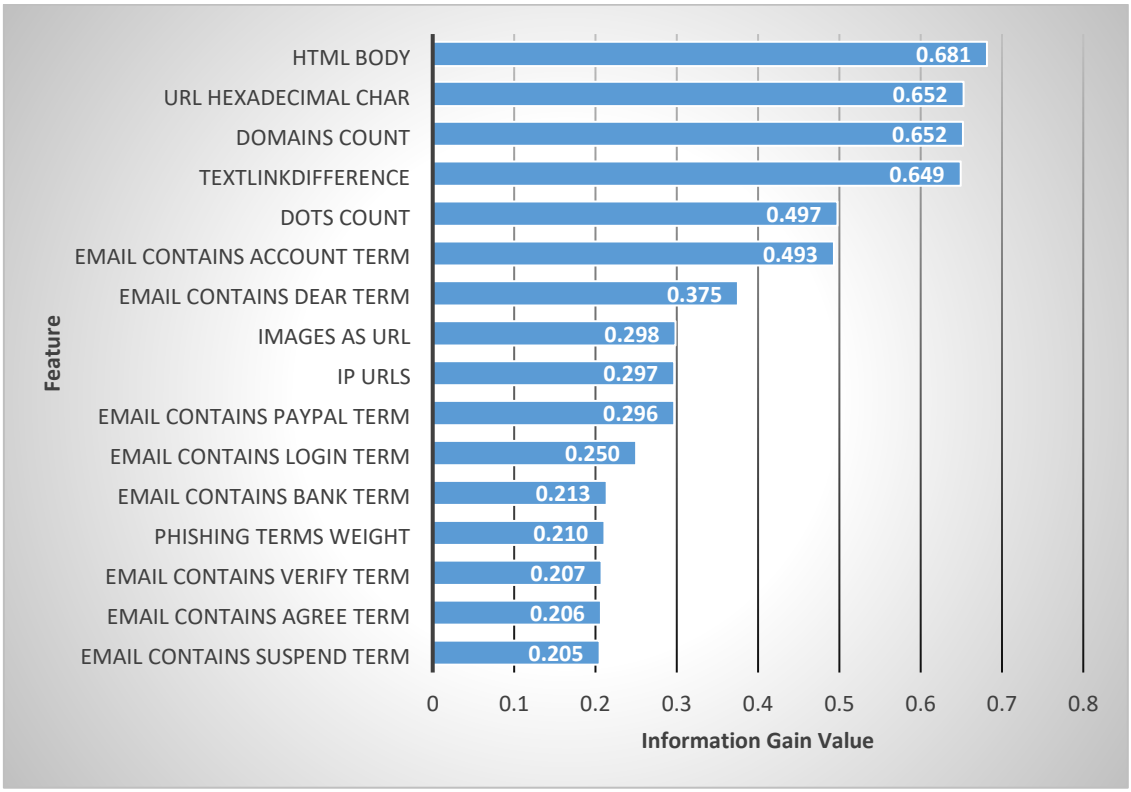

Figure 42: Features Information Gain Values

The pre-processing phase consists of a set of steps that utilizes the email header, body and text features to extract the features that contribute to the classification process, some features are extracted from the URL links in the email subject and body, such as

Hexadecimal URLs, Domains Count, TextLinkDifference, Dots Count, Images as URL and IP URLs. Other features are extracted from the email body such as HTML Body feature, the rest of features are extracted after processing the email subject and body text, this text processing step includes the following tasks:

- **Text parsing, tokenization and stemming**: the email subject and body text is parsed and tokenized into tokens, if the email body is HTML-formatted then the HTML tags are parsed to extract the text and identify URLs. Moreover, if the email contains attachments, they will also be parsed and tokenized. Each token in the extracted tokens is normalized such that morphological and in flexional endings of the tokens are removed, this stemming process is carried out using Porter Stemmer (PORTER, 1980).

- **Stop words removal**: in this step, extremely common words which would appear to be of little value are removed from the extracted tokens, common stop words include the tokens “the”,”then”,”he”,…etc. this step helps in reducing the similarities between emails and improves the performance of the proposed model specially in executing later steps.

- **Semantic text processing:** in this step, each token in the email is augmented with its conceptually-related words from the WordNet ontology (GEORGE A. MILLER, 1995) using the synonymy and hyponymy relationships, this step helps in identifying semantic relationships between tokens in different email messages and

thus shortening the distance between feature vectors that contain close proximity to one another, and hence enhances the classification accuracy.

- **Phishing terms weighting**: In this step, a set of phishing terms is built using the phishing emails in the training dataset. The phishing terms are those who have highest term frequency in the phishing data set. For example, the terms "Account" and "Please" existed in the phishing corpus 3384 and 3149 times respectively. This high frequency of terms indicates their importance in identifying phishing emails.

The proposed preprocessing model extracts the set of phishing terms- denoted by PT- from the set of phishing emails in the training data set, the phishing terms should also be not included in the legitimate emails training data set. The PT data set includes all terms whose document frequency (the number of phishing emails that contain the phishing term) is greater than 0.

Each term in the PT set is given a weight denoted by TW, and given by: $TW = \frac{TDFi}{N}$

Where TDFi is the term document frequency for term i in the PT data set, N is the number of phishing terms in the PT data. Table 9 depicts a sample of phishing terms and their respective document frequency and weight.

The phishing terms weight feature for each email is the sum of the weights of the phishing terms in that email, and given by $\sum_{i=1}^{n} TWi$, where n is the number of phishing terms in the email. The value of this feature indicates the weight of the phishing terms in that email.

Table 9: Sample phishing terms weights

Phishing Term	TDF	TW
Account	3384	2.256
Click	2550	1.7
PayPal	1172	0.781
Bank	1168	0.779
Passcode	20	0.013

The email's phishing terms weight feature could be calculated using the following pseudocode:

```
N is the number of phishing terms in the phishing
emails corpus.
T is the set of phishing terms in the email.
TW is an array that contains each phishing term weight.
W =0; //the phishing terms weight for the email.

For t in T loop
 W+ = TWt , where TWt is the weight of phishing term t.
End loop;
W=W/N;
```

6.5 Classification Model Building

After extracting the set of features from the training data set, we tested the classification accuracy of our model using five well known classification techniques; J48, Naïve Bayes, Support Vector Machine (SVM), Multi-Layer Perceptron and Random Forest. Before exploring the classification results for each algorithm, a brief summary of each algorithm's technique is presented as follows:

J48 algorithm: is the java implementation of the C4.5 classification algorithm, it uses a set of training data (S) consisting of already classified samples in the form $S=s_1, s_2, \ldots, s_n$. Each sample s in the training data set consists of k-dimensional vector $(x_1, x_2, \ldots., x_k)$, where x_k represents the feature value of that sample. The algorithm constructs a decision tree from the training data set, where each node of the tree is realized by the feature that most effectively splits its set of samples into subsets using the information gain value. The main advantages of decision trees are their simplicity to explain and interpret and take into account the features relationships and interactions, however they do not support online learning and require rebuilding the tree each time new samples exist.

Naïve Bayes Classifier: this classifier uses the Bayes rule of conditional probability and makes use of all the data features, and analyses them individually on the assumption that they are equally important and independent of each other. The advantages of this classifier is its simplicity and quick convergence, however it cannot learn about the interactions and relationships between the features in each sample.

SVM: Support Vector Machine is a supervised machine learning algorithm that is mostly used for classification tasks in addition to regression tasks. In SVM each data item is plot as a point in n-dimensional space (n is the number of features in each sample in the training set) and the algorithm mission is to find the best hyper-plane that divides the two classes. SVM classifies non-linearly separable data by transforming them into a higher dimensional space (using a kernel function) where a separating hyperspace exists. SVM is known for its accuracy and its ability to classify data that is not linearly separable. However, SVM is memory-intensive and hard to interpret.

Multi-Layer Perceptron (MLP): is a feed forward artificial neural network that consists multi layers (usually 3) of neurons, each neuron is considered a processing unit that is activated using an activation function. MLP is a supervised machine learning method in which the network is trained using a labelled training data set, a trained MLP will be able to map a set of input data (email features in our case) into a set of outputs (email class).

Random Forest: is decision tree based classification algorithm that is suitable for large data sets; it constructs a set of decision trees at training phase such that each tree operates on a predefined number of attributes chosen randomly. The classification process takes place by a majority vote of the results from each individual tree. Random Forest is trained on different parts of the training data set and aims at solving the problem of overfitting that is usually faced when using decision trees.

6.6 Performance metrics

In order to evaluate our proposed phishing email classification model using different classification techniques, we applied a set of evaluation metrics for each algorithm:

- **True Positive Rate (TP):** the percentage of phishing emails in the training data set that were correctly classified by the algorithm. Formally, if the number of phishing emails in the data set is denoted by P and the number of correctly classified phishing emails by the algorithm is denoted by N_p, then

  $$TP = \frac{Np}{P} \qquad (1)$$

- **True Negative Rate (TN):** the percentage of legitimate emails that were correctly classified as legitimate by the algorithm. If we denote the number of legitimate emails that were correctly classified as legitimate by N_L and the total number of legitimate emails as L, then

$$TN = \frac{Nl}{L} \qquad (2)$$

- **False Positive Rate (FP):** is the percentage of legitimate emails that were incorrectly classified by the algorithm as phishing emails. If we denote the number of legitimate emails that were incorrectly classified as phishing by N_f, and the total number of legitimate emails as L, then

$$FP = \frac{Nf}{L} \qquad (3)$$

- **False Negative Rate (FN):** the number of phishing emails that were incorrectly classified as legitimate by the algorithm. If we donate the number of phishing emails that were classified as legitimate by the algorithm by N_{pl} and the total number of phishing emails in the data set is denoted by P, then

$$FN = \frac{Npl}{P} \qquad (4)$$

- **Precision:** measures the exactness of the classifier; i.e. what percentage of emails that the classifier labeled as phishing are actually phishing emails, and it is given by:

$$Precision = \frac{TP}{TP+FP} \qquad (5)$$

- **Recall:** measures the completeness of the classifier results; i.e. what percentage of phishing emails did the classifier label as phishing, and is given by:

$$Recall = \frac{TP}{TP+FN} \qquad (6)$$

- **F-measure:** also known as F-score, and is defined as the harmonic mean of Precision and Recall, and given by:

$$F - measure = \frac{2*Precision*Recall}{Precision+Recall} \qquad (7)$$

- **Receiver Operating Characteristic (ROC) Area:** a metric that demonstrates the accuracy of a binary classifier by plotting TP against FP at various threshold values.

6.7 Classification results and discussion

This section presents the results that the proposed classification model achieved by applying the five proposed classification algorithms to the features extracted from the data set of 10538 emails including 5940 ham emails from spam assassin project (SPAM ASSASSIN) and 4598 spam emails from Nazario phishing corpus (J. NAZARIO). The generated features were fed to the five classifiers, namely J48, Bayes Net, SVM, MLP and Random Forest. To avoid overfitting, we used 10-fold cross validation technique which uses 0.9 of the training data set as data for training the algorithm and the remaining 0.1 of training data set for testing purposes, and repeat this division of the data set for training and testing for 10 times. The experiments were conducted using the open source WEKA data mining software (MARK HALL et al., 2009).

The results were evaluated using the performance metrics discussed in the previous section. Table 10 depicts the weighted average of classification results for each of the algorithms.

The results show that our model achieves high accuracy rates in classifying phishing emails, and outperforms similar proposed classification schemes as we will explain in the next section, thanks to the proposed pre-processing phase and feature reduction and evaluation process in the proposed model. The inclusion of features with high information gain values yielded a high influence in improving the classification results. A comparison of the different algorithms results is plotted in figure 43.

Table 10: Classification Algorithms Accuracy results (Weighted Average)

Metrics / Algorithm	TP	FP	Precision	Recall	F-Measure	ROC Area
J48	0.984	0.019	0.984	0.984	0.984	0.9863
Bayes Net	0.954	0.066	0.947	0.945	0.945	0.9717
SVM	0.969	0.039	0.97	0.969	0.969	0.9650
Random Forest	0.991	0.011	0.991	0.991	0.991	0.9988
MLP	0.977	0.026	0.977	0.977	0.977	0.9870

The best results were achieved by the Random Forest classification algorithm, due to their usage of tree ensembles that are capable of dealing with non-linear features that are correlated to each other, and its bagging mechanism enables it to handle very well high dimensional spaces as well as large number of training examples which fits to our proposed model.

The Random Forests algorithm builds a set of different decision trees for classification; to classify a new mail from the input dataset, Random Forest puts the new email's features vector down each of the trees in the forest, and then a classification is obtained from each of the trees, and a classification with the most votes is returned by the algorithm. The ROC area diagram in figure 44 shows the accuracy of the random forest algorithm in separating phishing emails form legitimate ones.

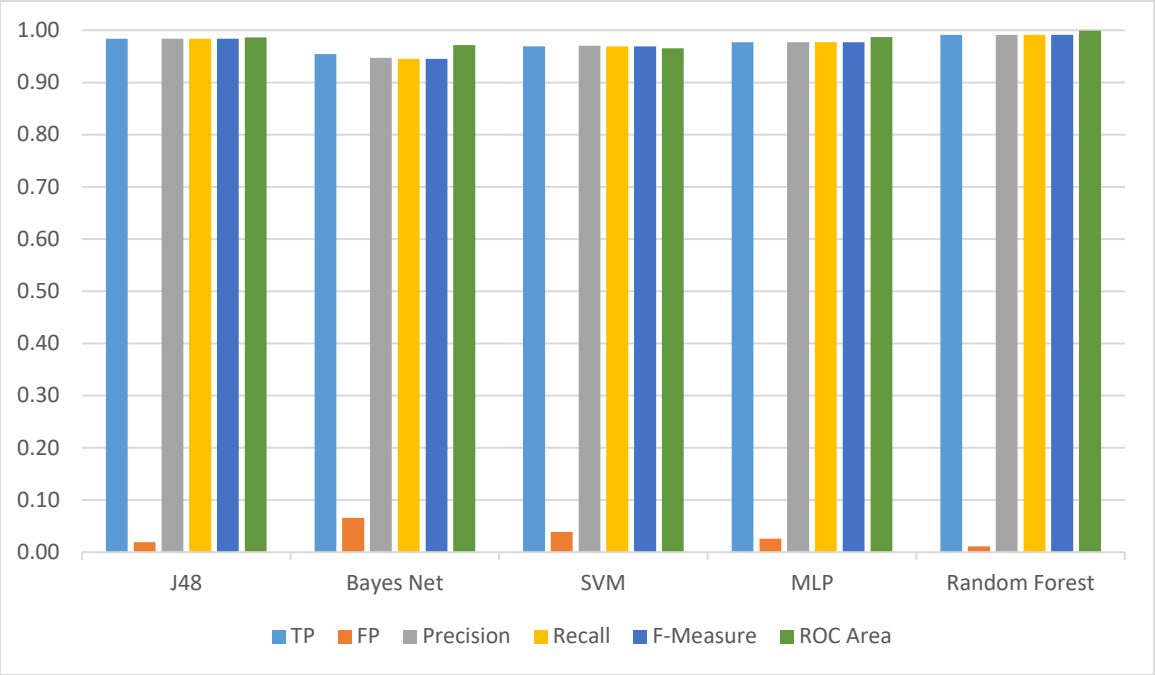

Figure 43: Classification results

We empirically evaluated the best number of trees to be used by random forest, the algorithm performed best when we set the number of trees to 30. The algorithm achieved

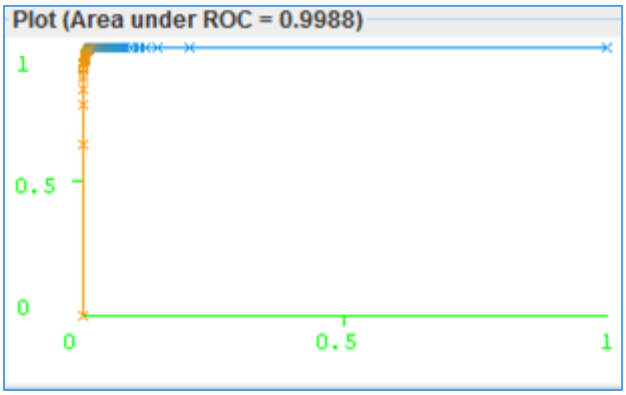

Figure 44: Random Forest ROC Area

0.988 accuracy and 0.014 FP rate when the number of trees was set to 10. Increasing the number of trees above 30 did not add a notable improvement to the classification results.

The J48 decision tree classifier achieved the second best classification results with 0.984 TP rate and 0.019 FP rate, and yielded a small enhancement over similar studies that implemented the same algorithms for phishing email detection such as the study in (SAMI SMADI et al., 2015). The J48 algorithm achieved 0.9863 ROC area accuracy as shown in figure 45.

The third best result was achieved using the MLP classifier with TP rate of 0.977 and 0.026 FP rate. The MLP achieved a ROC area of 0.987 as shown in figure 46. SVM and Bayes Net classifiers yielded a lower percentage of classification accuracy using the proposed feature set.

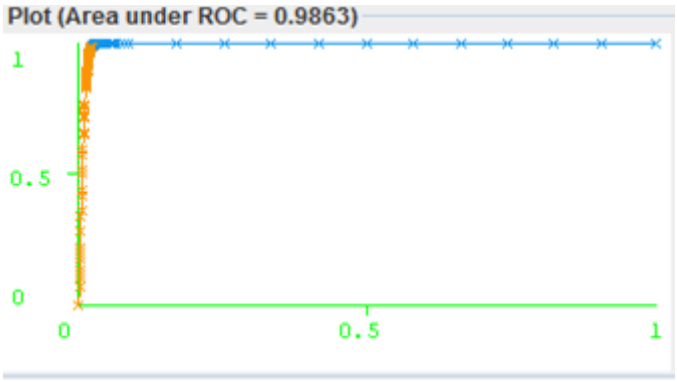

Figure 45: J48 ROC Area

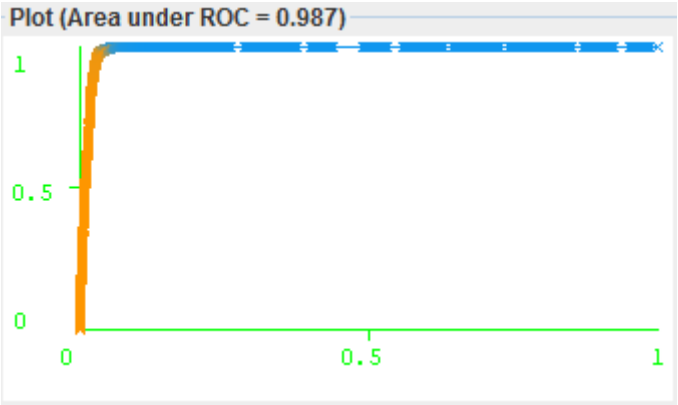

Figure 46: MLP ROC Area

6.8 Comparative Analysis

A set of proposed studies are found in the literature of phishing email detection using data mining techniques, in this section we compare our proposed model with a set of previously proposed models for phishing detection. Table 11 summarizes a set of seven previous related works along with the classification algorithm(s) used and the accuracy of the classification results, the results are visualized in figure 47.

The study in (M. KHONJI et al., 2012) used a feature vector of 47 features extracted from the same data sets of Nazario and Spam Assassin corpus, using Random Forest algorithm for training the classification model. Their model achieved 0.97 accuracy. Our model outperforms their model in accuracy rate with less feature set.

Table 11: Comparison of our approach with previous work

Paper Reference	Classification Algorithms	Accuracy
(M. KHONJI et al., 2012)	Random Forest	0.97
(W. N. GANSTERER AND D. P¨OLZ, 2009)	J48 + SVM	0.97
(M. CHANDRASEKARAN et al., 2006)	SVM	0.75
(L. MA et al., 2009)	decision trees, random forest, multi-layer perceptron, Naïve Bayes and SVM	0.99
(F. TOOLAN AND J. CARTHY, 2009)	C5.0	0.97
(I. R. A. HAMID AND J. ABAWAJY, 2011)	Bayes Net	0.96
(SAMI SMADI et al., 2015)	Random Forest, LibSVM, Bayes Net, SMO, Logistic Regression and NaiveBayes.	0.9811
Our Approach	J48, Bayes Net, SVM, Random Forest and Multi-Layer Perceptron.	0.991

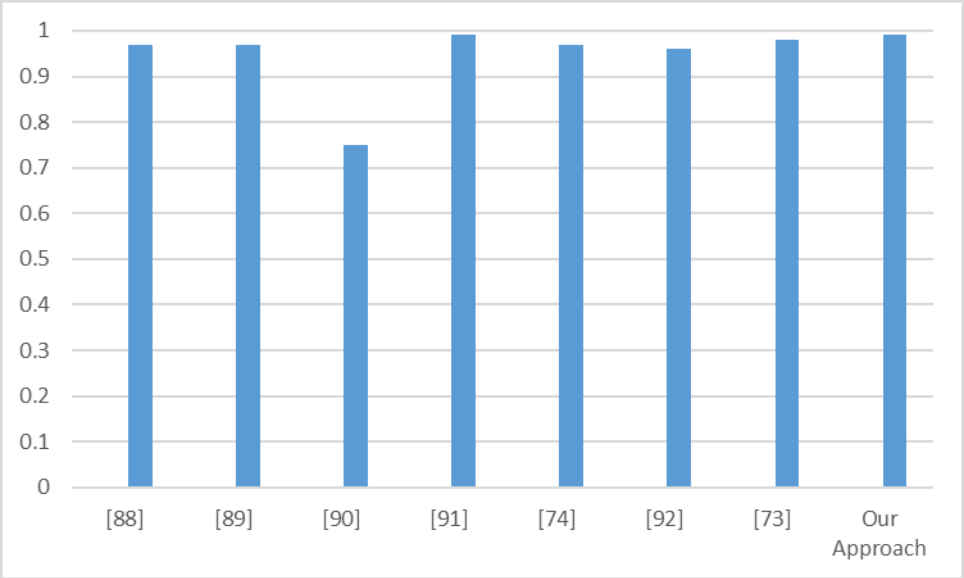

Figure 47: Comparison of our approach accuracy with related work.

The study in (W. N. Gansterer and D. P¨olz, 2009) applied both J48 and SVM for classifying emails using a feature set of 30 features and yielded an accuracy rate of 0.97, our approach outperforms this result using the same classification algorithm J48 with a classification accuracy of 0.984.

The study in (M. CHANDRASEKARAN et al., 2006) applied the SVM algorithm only on a feature set 25 features. The features were extracted from the email content only and achieved a low accuracy rate of 0.75, our model outperforms this result due to extracting features not only from the email body, but also from the header and also using the concept of phishing terms frequency.

The study in (L. MA et al., 2009) achieved high rate of accuracy in classifying phishing emails, it used a group of classification algorithms including Random Forest, Multi-Layer Perceptron, SVM and decision trees. However, this study was built on a small and not verified phishing data set.

The study in (F. TOOLAN AND J. CARTHY, 2009) achieved accuracy rate of 0.97 using the C 5 decision tree algorithm on a 22 features from two data sets of Ham, Spam emails. This result degraded to 0.84 when a third data set of Phishing emails was added.

The study in (I. R. A. HAMID AND J. ABAWAJY, 2011) achieved 0.96 accuracy using Bayes Net algorithm with seven hybrid features. However, this study was built over a small data set of 1645 emails, and when the data set was increased to 4594 emails the accuracy degraded to 0.92, and this is an indicator that their model has not been generalized.

The study in (SAMI SMADI et al., 2015) achieved an accuracy rate of 0.9811 and FP rate of 0.53 using the J48 algorithm and 23 hybrid features. Our approach enhances this result to accuracy of 0.984 using less features but with FP rate of 0.019 using J48, and accuracy of 0.991 and FP rate of 0.011using Random Forest. We believe that including only features that have IG values over the data set and introducing the feature of phishing terms weight for each email contributed to this enhancement in accuracy.

6.9 Summary and conclusions

This chapter proposed a classification model for emails into phishing or legitimate by applying the knowledge discovery and data mining techniques, the model was built using an intelligent pre-processing phase that extracts a set of features from the emails header, body and terms frequency. The features are enriched with WordNet ontology and text pre-processing technique of stemming to enhance the similarity between emails messages of a specific class. The extracted features were evaluated using the Information Gain measure and only those who have an information gain contribution were added to the feature set. Two accredited data sets were used in training and testing of the proposed model and 10-fold cross validation technique was used in the training and testing

processes to overcome the overfitting problem. The model was experimented using five popular data mining algorithms; Random Forest, J48, SVM, MLP and Bayes Net. The classification results achieved were encouraging and enhanced the classification accuracy so far registered in similar previously published models.

As future work, the proposed model could be further enhanced by developing an adaptive mechanism to reflect the contributions of analyzing new emails term frequency and applying enhanced linguistic processing techniques to strengthen the similarity between phishing emails terms such that a better classification results are obtained.

7 Conclusions and future work

The phishing problem is one of the cyber-attacks that threats people's security and privacy over the World Wide Web. Hundreds of thousands phishing attacks being conducted yearly aiming at compromising users' private data including their passwords and credit card numbers, attackers benefit from compromised users' data in a variety of ways including financial gain, identity hiding to perform criminal transactions, fame and notoriety and stealing industry secrets. Phishing attacks start by sending socially engineered messages to target users urging them to reveal their sensitive information, those message are usually written in a professional way so that they convince users that they originate from legitimate sources such as the institutions in which the users already have accounts. The phishing messages requests the user to fill in a form or visit a fake website that is professionally designed to be similar to the original institution's website, once the user reveals his private data, it is collected and compromised for the attacker's benefit.

Taking into consideration the life-cycle of a phishing attack, developing solutions to fight such attacks is not an easy task due to the set of factors that contribute in the design of such solutions including user security awareness and knowledge about phishing attacks, email level security and classification of possible phishing emails, website security and website authentication mechanisms.

In thesis, we worked on analyzing the phishing attacks lifecycle, and current defense mechanisms that aim at detecting or preventing phishing attacks before fooling the user to reveal his private information to the benefit of the attacker. Our work concentrated on understanding the anatomy of phishing attacks and current proposed solutions at various

levels, after that we contributed to the efforts of mitigating or preventing phishing attacks using two strategies:

1. Preventing phishing attacks by enhancing web authentication techniques: this strategy assumes that it is the website's vendor responsibility to protect the user's data and should never rely on user vigilance to fight phishing attacks, the proposed solutions under this strategy work on enhancing the web authentication techniques such they prevent phishing attacks from compromising the user's data and prevent attackers from doing transactions on behalf of the user even if the user was fooled and declared his private data. In this context, we proposed two enhanced authentication techniques, namely EARMAT and BMBAT; EARMAT works on the principle of extending the user authentication process by introducing a new level that enables the user to confirm or reject any login attempt to his account through the user's mobile device, this step employs the security principle of mutual authentication such both the web server and user mobile authenticate each other before the authentication process completes, this scheme fights phishing attacks as an attacker could not impersonate the user, as the attacker needs to not only compromise the user's password but also the he needs to compromise the second authentication factor, i.e. the user's mobile device to be able to access his account.

   Our second contribution in this strategy is BMBAT, an authentication scheme that replaces the current password-based authentication model with a new authentication model that leverages the user's mobile device as an identity prover for the user in the authentication process to any website. The proposed technique leverages the user's mobile, QR codes, PKI and symmetric keys to implement a

challenge-response model to verify the user identity against the server. This replacement of passwords in the authentication process neutralizes the phishing threat as the user does not rely on a password to be compromised by the attacker.

2. Mitigating phishing attacks by developing a smart phishing email classifier: the second strategy in our proposed solutions to phishing attacks is enhancing the phishing detection classifiers accuracy, in this context we built an email classifier that implements the knowledge discovery process and data mining techniques to identify the patterns in any email that could lead to identify that email as a legitimate or phishing email, the proposed email classifier employs the semantic text processing and ontologies to enhance the similarity detection process of email tokens such that different text tokens with similar meanings are processed together. Furthermore, the proposed classifier introduced the concept of phishing terms weighting such that each token in the email is checked against a list of known phishing terms and weighted accordingly, this weight values contributes to the classification process. At the end of this preprocessing phase, each email is represented using a set of 15 features that are fed to the classification model which decides whether it is a phishing or legitimate email. The proposed model was implemented using a group of well-known classifiers and achieved good results with classification accuracy of 99.1%.

   This classifier contributes to the process of fighting phishing attacks by being a defense line in the early stage of the phishing life cycle, such that it could block or at least notify the user about suspicious email messages that may lead to compromising the user's private data.

As future work, we suggest to enhance the proposed methods to fight phishing attacks as follows:

1. Enhancing the EARMAT authentication technique such that it supports different user accounts using the same mobile application.
2. Enhancing the BMBAT authentication technique to replace the PKI cryptographic algorithms with elliptic curve cryptography that needs far less processing power to run than RSA and symmetric keys, this enhancement opens the door for internet of things small devices to act as a part of the authentication scheme instead or side by side with user's mobile device.
3. Enhancing the proposed email classification model by incorporating multilingual ontologies such that the model generalizes to different languages than the English language.

8 References

A. ADAMS AND M. SASSE. (1999). “Users Are Not The Enemy,” Commun. ACM, vol. 42, no. 12, pp. 41–46.

A. AFROZ & R. GREENSTADT. (2011). PhishZoo: “Detecting Phishing Websites by looking at them”. In proceedings of the IEEE Fifth International Conference on Semantic Computing (ICS ’11).

A. ALNAJIM AND M. MUNRO. (2009). “An anti-phishing approach that uses training intervention for phishing websites detection,” in Proceedings of the 2009 Sixth International Conference on Information Technology: New Generations. Washington, DC, USA: IEEE Computer Society, pp. 405–410.

A. CZESKIS, M. DIETZ, T. KOHNO, D. WALLACH, AND D. BALFANZ. (2012). “Strengthening user authentication through opportunistic cryptographic identity assertions,” in Proceedings of the 2012 ACM conference on Computer and communications security, ser. CCS ’12, pp. 404–414.

A. PASHALIDIS AND C. J. MITCHELL. (2004). “Impostor: A single sign-on system for use from untrusted devices,” Proc. IEEE Globecom.

A. ROSIELLO, E. KIRDA, C. KRUEGEL, AND F. FERRANDI. (2007). “A layout-similarity based approach for detecting phishing Pages.” In IEEE International Conference on Security and Privacy in Communication Networks (SecureComm).

ALEXANDRA DMITRIENKO, CHRISTOPHER LIEBCHEN, CHRISTIAN ROSSOW, AHMAD-REZA SADEGHI. (2014). “Security Analysis of mobile two-factor authentication schemes”, Intel technology journal volume 18, Issue 4.

ANDROID CLIENT, available: https://developers.google.com/cloud-messaging/android/client , [Accessed July 2015]

ANTI PHISHING WORKING GROUP. ORIGINS OF THE WORD \PHISHING". Available: http://www.antiphishing.org/word_phish.html. [Accessed March 2015].
APWG REPORT. (2015). available: http://docs.apwg.org/reports/apwg_trends_report_q1-q3_2015.pdf [Accessed April 2016]

B. PARNO, C. KUO, AND A. PERRIG. (2006). “Phoolproof phishing prevention”, In Proceedings of the 10th International Conference on Financial Cryptography and Data Security (FC’06), pages 1–19.

C. WHITTAKER, B. RYNER, AND M. NAZIF. (2010). “Large-Scale Automatic Classification of Phishing Pages,”in Proceedings of the Network and Distributed System Security Symposium (NDSS).

CLOUDMARK. (2016). available: https://blog.cloudmark.com/2016/01/13/survey-spear-phishing-a-top-security-concern-to-enterprises/, [Accessed August 2016]

D. BALZAROTTI, M. COVA, AND G. VIGNA. (2008). “ClearShot: Eavesdropping on Keyboard Input from Video,” in IEEE Symp. Security and Privacy, pp. 170–183.

D. FLORÊNCIO AND C. HERLEY. (2008). “One-Time Password Access to Any Server Without Changing the Server,” ISC, Taipei.

D. L. COOK, V. K. GURBANI, AND M. DANILUK. (2008). “Phishwish: A stateless phishing filter using minimal rules,” in Financial Cryptography and Data Security, G. Tsudik, Ed. Berlin, Heidelberg: Springer-Verlag, pp. 182–186.

D. RECORDON AND D. HARDT. (2010). “The OAuth 2.0 Protocol,” April 2010, tools.ietf.org/html/draft-hammer-oauth2-00.

D. RECORDON AND D. REED. (2006) “OpenID 2.0: a platform for usercentric identity management,” in DIM ’06: Proc. 2nd ACM Workshop on Digital Identity Management, pp. 11–16.

DAVID PINTOR MAESTRE. (2012). “QRP: An improved secure authentication method using QR codes”. Available: https://www.grc.com/sqrl/files/QRP-secure-authentication.pdf. [Accessed January 2017]

DIVYA JAMES AND MINTU PHILIP. (2012). “A Novel Anti Phishing framework based on Visual Cryptography”. International Conference on Power, Signals, Controls and Computation. Pages: 1 - 5, DOI: 10.1109/EPSCICON.2012.6175228.

DODSON, B., ET AL. (2012). Secure, Consumer-Friendly Web Authentication and Payments with a Phone. Mobile Computing, Applications, and Services: Second International ICST Conference, MobiCASE 2010, Santa Clara, CA, USA, October 25-28, 2010, Revised Selected Papers. M. Gris and G. Yang. Berlin, Heidelberg, Springer Berlin Heidelberg: 17-38.

E. GAL´AN AND J.C. HERN´ANDEZ–CASTRO AND A. ALCAIDE AND A. RIBAGORDA . (2010). “A Strong Authentication Protocol based on Portable One–Time Dynamic URLs”, IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology.

F. TOOLAN AND J. CARTHY. (2009). “Phishing detection using classifier ensembles,” in eCrime Researchers Summit, 2009. eCRIME’09. IEEE, pp.1–9.

FACEBOOK CONNECT, available: https://developers.facebook.com/docs/facebook-login/overview. 2016 [Accessed January 2016]

G. GAFFNEY, “The myth of the stupid user,” available: http://www.infodesign.com.au/articles/themythofthestupiduser, [accessed March 2015].

GAURAV KUMAR TAK AND GAURAV OJHA. (2013). “MULTI-LEVEL PARSING BASED APPROACH AGAINST PHISHING ATTACKS WITH THE HELP OF KNOWLEDGE BASES”, International Journal of Network Security & Its Applications (IJNSA), Vol.5, No.6, November 2013

GEORGE A. MILLER (1995). WordNet: A Lexical Database for English. Communications of the ACM Vol. 38, No. 11: 39-41.

GOOGLE 2 STEP VERIFICATION, available: http://www.google.com/landing/2step/ [Accessed August 2015]

GOOGLE CLOUD MESSAGING, available: https://developers.google.com/cloud-messaging/gcm , [Accessed July 2015]

GOOGLE DEVELOPER CONSOLE, available: https://console.developers.google.com/ [Accessed July 2015]

GOOGLE SAFE BROWSING API, available: https://developers.google.com/safe-browsing/?hl=en [Accessed Nov. 2015]

GOOGLE STUDY. (2014). available: https://security.googleblog.com/2014/11/behind-enemy-lines-in-our-war-against.html [Accessed April 2016]

H. BAY, T. TUYTELAARS, AND L. VAN GOOL. (2006). "Surf: Speeded up robust features." European Conference on Computer Vision–ECCV 2006. Springer Berlin Heidelberg, 404-417.

H. HUANG, J. TAN, AND L. LIU. (2009). "Countermeasure techniques for deceptive phishing attack," in International Conference on New Trends in Information and Service Science, 2009. NISS '09, 2009, pp. 636 – 641.

I. R. A. HAMID AND J. ABAWAJY. (2011). "Hybrid feature selection for phishing email detection," in Algorithms and Architectures for Parallel Processing. Springer, pp. 266–275.

J. BONNEAU, C. HERLEY, P. C. V. OORSCHOT, AND F. STAJANO. (2012). "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," in Proceedings of the 2012 IEEE Symposium on Security and Privacy, pp. 553–567.

J. MASON. (2005) "The apache spamassassin public corpus", http://spamassassin.apache.org/publiccorpus/. Accessed June 2016.

J. NAZARIO, "Phishing Corpus", https://monkey.org/~jose/phishing/, Accessed June 2016.

J. S. DOWNS, M. HOLBROOK, AND L. F. CRANOR. (2007). "Behavioral response to phishing risk," in Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit, ser. eCrime '07. New York, NY, USA: ACM, pp. 37–44.

JULIE S. DOWNS, MANDY B. HOLBROOK, LORRIE FAITH CRANOR. (2006). "Decision Strategies and Susceptibility to Phishing", Symposium on Usable Privacy and Security (SOUPS), July 12-14, Pittsburgh, PA, USA.

L. MA, B. OFOGHI, P. WATTERS, AND S. BROWN. (2009) "Detecting phishing emails using hybrid features," in Ubiquitous, Autonomic and Trusted Computing, 2009. UIC-ATC'09. Symposia and Workshops on. IEEE, pp. 493–497.

LASTPASS, http://www.lastpass.com ,[Accessed August 2015]

LEW MAY FORM, KANG LENG CHIEW, SAN NAH SZEAND WEI KING TIONG. (2015). “Phishing Email Detection Technique by using Hybrid Features”, IT in Asia (CITA), 9th International Conference.

LITAN, A. (2004), “Phishing Attack Victims Likely Targets for Identity Theft”. Gartner Research.

LOFTESNESS S. (2004). Responding to "Phishing" Attacks. Glenbrook Partners.

M. CHANDRASEKARAN, K. NARAYANAN, AND S. UPADHYAYA. (2006). “Phishing email detection based on structural properties,” in NYS Cyber Security Conference, pp. 1–7.

M. KHONJI, Y. IRAQI, AND A. JONES. (2012). “Enhancing phishing e-mail classifiers: A lexical url analysis approach,” International Journal for Information Security Research (IJISR), vol. 2, no. 1/2.

M. WU, R. C. MILLER, AND S. L. GARFINKEL. (2006). “Do security toolbars actually prevent phishing attacks?” in Proceedings of the SIGCHI conference on Human Factors in computing systems, ser. CHI ’06, New York, NY, USA, pp. 601–610.

MARK HALL, EIBE FRANK, GEOFFREY HOLMES, BERNHARD PFAHRINGER, PETER REUTEMANN, IAN H. WITTEN (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

MARKUS JAKOBSSON and STEVEN MYERS. (2007). “Phishing and countermeasures: understanding the in-creasing problem of electronic identity theft”. John Wiley & Sons, Inc.

MAYANK PANDEY AND VADLAMANI RAVI. (2012) “Detecting phishing e-mails using Text and Data mining”, IEEE International Conference on Computational Intelligence and Computing Research.

MENGJUN XIE, YANYAN LI, KENJI YOSHIGOE, REMZI SEKER, JIANG BIAN. (2015). “CamAuth: Securing Web Authentication with Camera”, IEEE 16th International Symposium on High Assurance Systems Engineering.

METE EMINAGAOGLU, ECE CINI, GIZEM SERT AND DERYA ZOR. (2014). “A Two-Factor Authentication System with QR Codes for Web and Mobile Applications”. Fifth International Conference on Emerging Security Technologies. DOI 10.1109/EST.2014.19.

MOZILLA FIREFOX, available: www.mozilla.org/. [Accessed August 2015]

N. CHOU, R. LEDESMA, Y. TERAGUCHI, AND J. C. MITCHELL. (2004). “Client-side defense against web-based identity theft,” in NDSS. The Internet Society.

P. KUMARAGURU, Y. RHEE, A. ACQUISTI, L. F. CRANOR, J. HONG, AND E. NUNGE. (2007). “Protecting people from phishing: the design and evaluation of an embedded training email system,” in Proceedings of the SIGCHI conference on Human

factors in computing systems, ser. CHI '07. New York, NY, USA: ACM, 2007, pp. 905–914.

P. LIKARISH, D. DUNBAR, AND T. E. HANSEN. (2008). "Phishguard: A browser plug-in for protection from phishing," in 2nd International Conference on Internet Multimedia Services Architecture and Applications, 2008. IMSAA 2008, pp. 1 – 6.

P. PRAKASH, M. KUMAR, R. R. KOMPELLA, AND M. GUPTA. (2010). "Phishnet: predictive blacklisting to detect phishing attacks," in INFOCOM'10: Proceedings of the 29th conference on Information communications. Piscataway, NJ, USA: IEEE Press, pp. 346–350.

PAL, N.R., JAIN, L.C., (EDS.). (2005) "Advanced Techniques in Knowledge Discovery and Data Mining", Springer Verlag.

PHISHING SAMPLE, available: http://lts.lehigh.edu/phishing/examples/online-banking , [Accessed February 2016].

PORTER, M.F. (1980), "An algorithm for suffix stripping", Program, Vol. 14 No.3, pp. 130-137.

PRAJAKTA OZARKAR, & DR. MANASI PATWARDHAN. (2013)." Efficient Spam Classification by Appropriate Feature Selection", International Journal of Computer Engineering and Technology (IJCET), ISSN 0976 – 6375(Online) Volume 4, Issue 3, May – June.

PUSHY, available: https://pushy.me/ , [Accessed July 2015]

R. MORRIS AND K. THOMPSON. (1979). "Password security: a case history," Commun. ACM, vol. 22, no. 11, pp. 594–597.

R.S. RAO, AND S.T. ALI. (2015) "A Computer Vision Technique to Detect Phishing Attacks". In Fifth International Conference on Communication Systems and Network Technologies.

RACHNA DHAMIJA, J. D. TYGAR, MARTI HEARST. (April 2006). "Why Phishing Works", CHI-2006: Conference on Human Factors in Computing Systems.

S. ABU-NIMEH, D. NAPPA, X. WANG, AND S. NAIR. (2007). "A comparison of machine learning techniques for phishing detection," in Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit, ser. eCrime '07. New York, NY, USA: ACM, pp. 60–69.

S. EGELMAN, L. F. CRANOR, AND J. HONG. (2008). "You've been warned: an empirical study of the effectiveness of web browser phishing warnings," in Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems, ser. CHI '08. New York, NY, USA: ACM, pp. 1065–1074.

S. GAW AND E. W. FELTEN. (2006). "Password Management Strategies for Online Accounts," in ACM SOUPS 2006: Proc. 2nd Symp. on Usable Privacy and Security, pp. 44–55.

S. GORLING. (2006) “The Myth of User Education,” Proceedings of the 16th Virus Bulletin International Conference.

S. SHENG, B. WARDMAN, G. WARNER, L. F. CRANOR, J. HONG, AND C. ZHANG. (July 2009). “An empirical analysis of phishing blacklists,” in Proceedings of the 6th Conference in Email and Anti-Spam, ser. CEAS’09, Mountain view, CA.

S. SHENG, M. HOLBROOK, P. KUMARAGURU, L. F. CRANOR, AND J. DOWNS. (2010). “Who falls for phish?: a demographic analysis of phishing susceptibility and effectiveness of interventions,” in Proceedings of the 28th international conference on Human factors in computing systems, ser. CHI ’10. New York, NY, USA: ACM, pp. 373–382.

S.H. KIM, S.H. LEE AND S.H. JIN. (2013). “Active Phishing Attack and its Countermeasures”, Electronics and Telecommunications Trends, Vol 28, No3, pp. 30-50.

SAMI SMADI, NAUMAN ASLAM, LI ZHANG, RAFE ALASEM, M A HOSSAIN. (2015). “Detection of Phishing Emails using Data Mining Algorithms”, 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA).

SPAM ASSASSIN, available: http://spamassassin.apache.org/ [Accessed March. 2016]

SPOOFGUARD. Client-side defense against web-based identity theft. http://crypto.stanford.edu/ SpoofGuard/, 2005.

SUNIL B. RATHOD AND TAREEK M. PATTEWAR. (2015). “Content Based Spam Detection in Email using Bayesian Classifier”, IEEE ICCSP conference.

SYMANTIC INTELLIGENCE REPORT, available: https://www.symantec.com/content/en/us/enterprise/other_resources/intelligence-report-06-2015.en-us.pdf [Accessed May 2016]

T. MOORE AND R. CLAYTON. (2007). “Examining the impact of website take-down on phishing,” in eCrime ’07: Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit. New York, NY, USA: ACM, pp. 1–13.

TAREEK M. PATTEWAR, SUNIL B. RATHOD. (2015). “A Comparative Performance Evaluation of Content Based Spam and Malicious URL Detection in E-mail”, IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS).

TRUSTZONE, available: https://www.arm.com/products/security-on-arm/trustzone. [Accessed November, 2016]

TWO FACTOR AUTHENTICATION, available: http://www.google.com/patents/US4720860. [Accessed August, 2015]

USAMA FAYYAD, GREGORY PIATETSKY SHAPIRO AND PADHRAIC SMYTH. (1996). “Knowledge Discovery and Data Mining: Towards a Unifying Framework”, KDD-96 Proceedings.

W. D. YU, S. NARGUNDKAR, and N. TIRUTHANI, (July 2008) “A phishing vulnerability analysis of web based systems,” in Proceedings of the 13th IEEE Symposium on Computers and Communications (ISCC 2008). Marrakech, Morocco: IEEE, pp. 326–331.

W. N. GANSTERER AND D. P¨OLZ. (2009). “E-mail classification for phishing defence”, in Advances in Information Retrieval. Springer, pp. 449–460.

X. DONG, J. CLARK, AND J. JACOB. (May 2008). “Modelling user-phishing interaction”, in Human System Interactions, 2008 Conference, pp. 627–632.

Y. ZHANG, J. I. HONG, AND L. F. CRANOR. (2007). “Cantina: a content-based approach to detecting phishing web sites,” in Proceedings of the 16th international conference on World Wide Web, ser. WWW ’07. New York, NY, USA: ACM, pp. 639–648.

9 ملخص الرسالة باللغة العربية

التصيد هو نوع من هجمات الاحتيال عبر الإنترنت التي توظف رسائل إلكترونية إحترافية لخداع المستخدمين و دفعهم الى التصريح بمعلومات حساسة خاصة بهم كإسم المستخدم وكلمة المرور. هجمات التصيد تبدأ من خلال التواصل مع مستخدم أو مجموعة من المستخدمين بإستخدام رسائل البريد الإلكتروني المهنية (أو من خلال مكالمة هاتفية أو رسالة نصية..الخ) بحيث توهم المستخدم بأنها من مصدر شرعي (مثل المؤسسات التي يتعامل معها المستخدم)، وتطلب هذه الرسالة من المستخدم إدخال أو تغيير كلمة المرور الخاصة به من خلال موقع إلكتروني مزيف على شبكة الإنترنت تم تصميمه بشكل إحترافي ليكون مشابه للموقع الأصلي للمؤسسة، و هذا النوع من الهجمات الإلكترونية عادة ما تتكيف طبيعتها مع التدابير الأمنية المضادة التي يتم إتخاذها من قبل مزودي الخدمات على شبكة الإنترنت و المستخدمين أنفسهم على حد سواء، وبذلك أصبحت تشكل تهديداً حقيقياً لأمن المؤسسات المالية ومواقع التجارة الإلكترونية ومواقع الإنترنت بشكل عام.

في هذه الأطروحة، سوف نناقش ونحلل هجمات التصيد والحلول وآليات الدفاع التي يتم تطبيقها لتخفيف أو منع وقوع هجمات التصيد، وسيركز العمل المقترح لمكافحة هجمات التصيد على إستخدام إستراتيجيتين اثنتين؛ وسوف تركز الإستراتيجية الأولى على منع هجمات تصيد المعلومات عن طريق مقترحات لحل نقاط الضعف في أنظمة مصادقة الولوج الى مواقع الويب الحالية، و في هذا السياق، تناقش الأطروحة مقترحين جديدين لأنظمة المصادقة المضادة لهجمات التصيد، الإقتراح الأول يعتمد على إضافة مستوى جديد في عملية المصادقة يوظف الهاتف المحمول للمستخدم كعامل مصادقة ثاني بالإضافة الى إسم المستخدم و كلمة المرور، بحيث يمكن المستخدم من التحكم في أي محاولة تسجيل دخول على حسابه من خلال تطبيق ذكي من جهاز الهاتف الخاص به، و يتم تنبيه المستخدم عن أي محاولة دخول الى حسابه بالإعتماد على تقنية المراسلة السحابية المقدمة من شركة جوجل و التي توفر إمكانية تنبيه المستخدم بشكل فوري عن محاولة الدخول الى حسابه، و يوظف

النظام تقنية المصادقة المتبادلة المبنية على التشفير بإستخدام المفتاح العمومي للتأكيد المتبادل على هوية خادم موقع الإنترنت و هوية التطبيق الذكي على جهاز المستخدم.

أما نظام المصادقة الثاني المقترح فيعمل على مبدأ إلغاء المصادقة على الدخول باستخدام مبدأ كلمة المرور التقليدية، بحيث يتم استخدام تطبيق ذكي على الهاتف المحمول الخاص بالمستخدم ليقوم بمهام المصادقة و يمثل الهوية الإلكترونية للمستخدم، يوظف هذا المقترح مبدأ التواقيع الرقمية و المفتاح المشترك المتماثل و تقنية رمز الاستجابة السريعة للإستعاضة عن كلمة المرور برمز دخول آلي يتم بناؤه و تشفيره على مرحلتين باستخدام المفتاح المشترك المتماثل ومن ثم باستخدام المفتاح العمومي من قبل الخادم عند محاولة المستخدم الولوج الى الموقع الإلكتروني، ويتم إدماج هذا الرمز ضمن رمز الإستجابة السريعة و يعرض على متصفح المستخدم بحيث تتم قرائته إلكترونياً من قبل التطبيق الذكي على هاتف المستخدم، وبدوره يقوم التطبيق بالتأكد من هوية الخادم و من ثم عرض هوية المستخدم للخادم بشكل تلقائي في حال توفر اتصال بشبكة الانترنت او عرض جزء من رمز الدخول للمستخدم بحيث يمكنه استخدامه كرمز دخول لمرة واحدة لإستكمال عملية المصادقة و الدخول الى الموقع الالكتروني.

الإستراتيجية المقترحة الثانية في مكافحة هجمات تصيد المعلومات تركز على التخفيف من هجمات تصيد المعلومات عن طريق بناء تقنية تصنيف البريد الإلكتروني و إكتشاف رسائل التصيد وإتخاذ الإجراءات اللازمة لمنع عرضها للمستخدم على مستوى نظام البريد الإلكتروني، وتعتمد تقنية التصنيف المقترحة على تطبيق مبادئ إكتشاف المعرفة، تقنيات استخراج البيانات وتقنيات معالجة النص الدلالي، وقد تم إختبار تقنية التصنيف المقترحة على مجموعتين من البيانات المعتمدة تتألفان من أكثر من 10000 رسالة بريد الكتروني مختلطة تضم رسائل مشروعة ورسائل تصيد، وحققت التقنية معدل تصنيف إيجابي بنسبة 99.1٪ باستخدام خوارزمية الغابات العشوائية.