Journal of Information Security and Applications 42 (2018) 1–17

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Secure authentication scheme to thwart RT MITM, CR MITM and malicious browser extension based phishing attacks

Gaurav Varshney[a,*], Manoj Misra[a], Pradeep Atrey[b]

[a] Department of CSE, IIT Roorkee, Roorkee, Uttarakhand, India
[b] Computer Science Department, State University of New York at Albany, NY, USA

ARTICLE INFO

Article history:
Available online 2 August 2018

Keywords:
Phishing
Malicious browser extension
Bluetooth
RT MITM
CR MITM
Authentication
Chrome
Smartphone
Credentials
Deception
Multifactor authentication
QR code
OTP
CAPTCHA

ABSTRACT

Securing user credentials against phishing attacks is an important and challenging research problem. These days phishing is carried out by real time (RT) and control relay (CR) man in the middle (MITM) attacks or by malicious browser extensions. Existing user authentication schemes are either incapable of handling these attacks or they are complex to learn and use or they require users to purchase and carry additional hardware such as a security key. In this paper, we propose a new secure authentication scheme for anti-phishing, which uses the Bluetooth address of the user's smartphone for user identification along with App instance ids and a user password for authentication. The analysis of the results of our experiments shows that the proposed scheme is safe against RT MITM and CR MITM phishing attacks and the attacks launched via malicious browser extensions. It is also efficient in terms of memory and CPU utilization. The comparison of the proposed scheme with the existing schemes in terms of usability and deployability shows that it is better than the schemes that can provide the same level of security.

1. Introduction

Phishing [1–16] is a cyber-fraud which uses deception techniques to break secure authentication schemes. Most of the existing authentication schemes, either one way (user authenticates himself to the website) or two way (the website and user both authenticate each other), are vulnerable to the latest cyber phishing attacks [14,15,17] such as Real Time (RT) and Control Relay (CR) Man In The Middle (MITM) phishing attacks [18–21] and malicious browser extension based phishing (MEP) attacks [22–28]. MEP attacks generally involve keylogging, password & form data sniffing, screen logging etc. The consistent number of phishing attacks [29–32] can be inferred as an indirect indication of the ease with which existing authentication schemes can be compromised.

- *RT MITM Phishing:* In an RT MITM [18–21] phishing attack, attackers place themselves between a client and a server by means of a phishing website appearing as a genuine one. The attacker captures the authentication information entered by the user on the phishing website and relays this information to the genuine website in real time via manual or automated means, thereby gaining access to the user's account. RT MITM, unlike traditional phishing, can utilize remote desktop monitoring modules, malicious browser extensions/screen loggers that can help in providing the additional information (keystrokes, CAPTCHAs, QR codes etc.) in real time to break the authentication scheme [Appendix]. QRLjacking [33] is an example of an RT MITM phishing attacks.
- *CR MITM Phishing:* CR MITM [19–21] is more invasive. In a CR MITM phishing attack, the attacker relays his desktop over the user's terminal, eventually deceiving the user into entering his credentials directly on his computer. Both one-way and two-way authentication [34,35] schemes are vulnerable to such attacks as an authentication token provided by the user can be captured on the phishing website and can be relayed to the genuine website in real time to complete a successful authentication. Only separate hardware token based schemes or schemes which store at least one part of user credentials over the client can handle such attacks.
- *MEP Attacks*: Malicious browser extensions [36–41] can also be used to perform phishing attacks for stealing user credentials. Schemes for malicious browser extension detection have been proposed in the past [22–28,36,41–43], but little work has been

* Corresponding author.
E-mail addresses: gauravdtsi@gmail.com (G. Varshney), manojfec@iitr.ac.in (M. Misra), patrey@albany.edu (P. Atrey).

https://doi.org/10.1016/j.jisa.2018.07.001
2214-2126/

done in this area. Malicious browser extensions can acquire permissions needed for carrying out any stealth activity by providing a functionality to the users in the foreground that requires the same set of permissions. For example, a malicious browser extension can provide grammar checking facility to the users in the foreground and hence can get permissions to access the contents of websites opened in the browser and tab related data (URL typed in the tab's address bar). Using these permissions malicious browser extensions can carry out credential stealing, spying and phishing activities in the background. Malicious browser extensions can also be installed covertly by an insider on the victim's PC. A malicious browser extension can perform key logging, screen logging, or password sniffing to steal credentials. No matter whether the scheme is a CAPTCHA based scheme [20,44], a picture password-based scheme [45], or a dynamic security skin based scheme [46], it can be compromised if a malicious browser extension running on a user's PC captures the screen and relays this information to the attacker in real time. A malicious browser extension can sniff the information entered on the browser even before the application or transport layer (TLS) encrypts it, hence password manager based schemes can also be compromised. In one of our recent publications, we have discussed in detail the attacks which can be launched via malicious browser extensions [40]. Also, see the [Appendix] for MEP attack examples.

Most of the existing multifactor [47] authentication schemes are incompetent in handling the attacks described above. Our experiments show that OTP/PIN-based schemes [48–50], QR/Barcode based schemes [19,21], Password manager [51], and push notification based login schemes [52–54] are vulnerable to these attacks. Graphical password-based schemes [20,44,55] can be phished using a malicious browser extension that can log the screen when the user enters his password or CAPTCHA on the website opened in the browser. Also, graphical password based schemes are not user-friendly [56]. Biometric authentication [57–59] is still not 100% accurate, robust, mature, and user-friendly. Environment and usage can affect the measurements and they also need additional hardware [60]. User-friendly biometric schemes which are commercialized (such as fingerprints, facial recognition etc.) can be spoofed [59,61,62]. Separate hardware token based schemes (such as Yubikey U2F [63], RSA SecurID [64], DUO [65] etc.) provide a better layer of security compared to the other schemes but they have following drawbacks.

- Firstly, the user needs to buy and carry these hardware tokens always which makes them nonuser friendly [50].
- Secondly, some of the hardware token based schemes that use security keys for OTP/PIN generation and their subsequent entry on browsers can be compromised via malicious browser extensions through sniffing of HTML form data during its submission.
- Thirdly security keys such as Yubikey and RSA SecurID tokens can also be compromised through reverse engineering and spoofing onto other hardware [66].

New protocols such as Yubikey U2F may handle most of the sophisticated attacks but the need for buying and carrying a separate authentication token makes them unattractive. Also, attacks that weaken the strength of RSA key generation on Yubikey has been recently recorded in October 2017 [67]. This can be inferred from the ratio of the number of users who use separate hardware authentication tokens to log in over websites to the number of users who use soft tokens generated from Authenticator Apps installed on their smartphones or OTPs as a second factor for authentication [50].

Hence there is a need for immediate research and development of secure authentication schemes which can address the latest phishing threats from the latest cyber phishing attacks. The schemes should also be easy to use and must use existing hardware and/or technology so that the cost incurred for carrying out login authentication can be reduced. In this paper, we propose a secure authentication scheme which uses a commonly used hardware, smartphone to provide better security against the latest phishing threats. The main contributions of the paper include:

1. We analyze the security of existing popular multifactor authentication schemes against latest phishing attacks launched via RT MITM, CR MITM and malicious browser extensions. Our analysis shows that most of these schemes can't handle these attacks. Only some of the schemes such as Yubikey U2F, Tricipher [68] can handle these attacks but they require the user to purchase and carry extra hardware device.
2. We propose a secure authentication scheme that can handle RT MITM, CR MITM and malicious browser extension based phishing attacks and uses a smartphone, a device commonly used by the Internet users.

Section 1 introduces the latest phishing attacks and establishes the motivation to work in this area. It also describes the main contributions of the paper and its organization. Section 2 describes latest and popular multifactor authentication schemes that are currently used by the websites for authentication. Recent proposals which claim to handle RT MITM and CR MITM phishing attacks have also been described. The section ends with a discussion of research gaps in this area. Section 3 explains the design and working of the proposed secure authentication scheme that addresses the research gaps identified in section 2. Section 4 provides information regarding the implementation, performance, security evaluation of the proposed scheme and a comparison with existing schemes in terms of usability, deployability and security. Section 5 mentions the key limitations of the proposed scheme and concludes the paper. The scope for future work is also given in the section.

2. Literature survey

2.1. Existing multifactor authentication schemes

Two Factor Authentication via OTP /PIN: Google 2 step [49,69] verification is a two-factor authentication scheme which uses OTP as a second factor. The server sends OTP to the user's registered mobile number after receiving the user credentials. OTP, if entered correctly by the user, allows him to login onto the website. SAASPASS [48,70] is another two-factor authentication scheme which uses App generated PINs in place of SMS based OTPs. This reduces the cost of sending OTPs at every login. The user installs the SAASPASS App on his/her smartphone and links it with his/her personal web account. SAASPASS generates and displays a 6-character PIN to the user which is synchronized with the server and changes every 30 seconds. The user enters the PIN as the second factor for login verification. RSA Soft token, DUO also generate similar authentication code/PIN through Apps for login. These schemes are vulnerable to MITM phishing attacks as the OTP/PIN can be acquired by a phishing website or through a malicious browser extension [Appendix].

Authentication using QR Codes: In Xie et al.'s [21] approach, a user submits the username and password to the website using a browser extension. The server generates and sends a barcode to the user. This barcode is displayed on the user's browser. The user scans the barcode using his smartphone App and after verification generates a vouch request in the form of a barcode which is scanned by the PC camera. The browser extension sends the vouch request to the server for final authentication. The approach claims to solve the problem of MITM phishing attacks and utilizes

Diffie-Hellman to secure the communication channel between the user's browser and the server. Kim et al. [19] proposed an approach to provide security against MITM phishing attacks. Their approach uses QR codes to exchange user credentials. It additionally uses the IP address of the Smartphone to verify the proximity of the user and the PC used for login. However, IP addresses can be spoofed which can cause the compromise of the scheme. The scheme proposed by Mukhopadhyay et al. [71] uses a third-party verifier. Both the third party and the user's Smartphone share a secret key. The third party checks the user credentials (submitted over the website) and after verification sends a challenge in an encrypted (using the shared secret key) QR code to the user. The user scans the QR code using his Smartphone App, verifies it, and sends back the encrypted response to third-party verifier. After verifying the response, third-party gives the user access to the server. In Dodson et al.'s [72] scheme, both the user and the server share a secret key which is used for authentication. The server sends a challenge to the user in the form of a QR code. The user's smartphone App encrypts the server challenge using the shared secret key and sends it back to the server. The server verifies the response and allows the user to log in. During our experiments, we found that none of these schemes are secure against the attacks described in the previous section.

Authentication using graphical password and CAPTCHA: Leung et al [20] proposed the concept of flash-based OTP CAPTCHA to avoid sophisticated attacks such as MITM and MEP attacks where attackers can capture the user's screen to steal credentials. The OTP CAPTCHA has moving digits. The user's mouse click coordinates and the time of click are sent to the server. Based on this information the server identifies the digits of the OTP, selected by the user. The scheme is not user-friendly and also fails in case of CR MITM attack. The scheme proposed by Zhu et al. [44] uses CAPTCHA as a graphical password. The user clicks his password characters displayed in the CAPTCHA and the coordinates of the mouse click are used by the server to verify the password. The scheme fails to handle MITM and screen logging attacks.

Authentication using push notification: In push notification based authentication, the user enters a user identification token (username) on the website while initiating a login session. Once the server receives the username from the user it generates a push notification message and sends it to the registered App running on the user's Smartphone. After the user approves the push notification message received on his App, the server allows the user to log in. Such push notification based login is provided by popular organizations such as Yahoo, Google etc. These schemes can be compromised using MITM attacks described in Appendix.

Authentication using separate hardware tokens: Schemes that require users to buy an additional dedicated hardware (Such as USB security keys, Smart cards [73] etc.) for user authentication are considered as separate hardware token based schemes. Separate hardware tokens either store some cryptographic keys, passwords etc. which are communicated during the authentication process or the hardware tokens generate OTPs/PINs to be entered as the second factor during the authentication. The double armored Tricipher scheme [68] uses multipart credentials. One part of the credentials remains with the user whereas the other part is stored in a secure appliance kept in the enterprise data center. Moreover, a secret key is stored on the user's machine and is also known to the server. This key is used to encrypt the username and password entered by the user. The secure appliance signs (encrypts) the user credentials using the part of the credentials stored on it and sends them back to the user. The user sends these encrypted credentials directly to the server. This completes the authentication process. The triple armored Tricipher scheme requires an additional hardware security key during the authentication process. Other hardware token-based authentication schemes include RSA SecurID hardware tokens, U2F security keys such as Yubikey etc. RSA SecurID generates authentication code usually every 60 seconds using a clock and a random seed. The seed is provided to the token via the RSA server when the device is purchased. The server verifies the authentication code during the login by computing the authentication code which is valid at that moment using the seed stored in its records for the token and the value of the clock. If the code matches the user gets authenticated. Yubikey follows the U2F protocol for user verification during web authentication.

Password managers: Password managers [51] also provide an ease to user authentication. They store user credentials for individual websites and automatically enter them when the websites get visited by the user. Most of the password managers store the user credentials in an encrypted form in the browser storage and auto-fill them whenever the respective websites are opened in the browser. Password manager based schemes are vulnerable to MEP attacks. Other similar schemes include Ross et al. [74] where the authors used a browser extension to modify the password entered by the user, using the SALT stored at the client machine and the domain information of the website.

2.2. Analysis and research gaps

2.2.1. Security against MITM and MEP attacks

1. OTP and PIN-based schemes are vulnerable to RT MITM phishing as attackers can deceive users to enter OTP on phishing websites along with other credentials [Appendix]. One should also note that in special publication 800-63-3: digital authentication guidelines the National Institute of Standards and Technology (NIST) deprecated the use of SMS in 2FA services [75]. The OTP based 2FA can be compromised using malware such as Eurograbber [76].
2. QR code-based schemes are also vulnerable to MITM phishing attacks. An attack scenario illustrating how QR Code can be relayed to hack a WhatsApp web account is shown in the video link given in Appendix. QR code cloning on phishing website (also called QRLjacking [33]) to hijack user accounts makes QR code scanning based schemes vulnerable to RT MITM phishing attacks.
3. Graphical password-based schemes can be compromised using CR MITM attacks. Also, malicious browser extensions can log browser's tab screen to get the information of user mouse clicks to predict the graphical input.
4. Push notification based login schemes are vulnerable to RT MITM phishing attacks as the username can be obtained on a phishing website and can be used to initiate a push notification by the attacker through his browser session that user will approve in deception. This attack scenario is discussed in the Appendix in detail. Other than this push notification for login can also be abused to launch phishing attacks on users [77].
5. RSA SecurID soft token and hardware tokens that generate a PIN can be compromised via sniffing passcode through malicious browser extensions. Few separate hardware token based schemes such as Yubikey U2F are secure from RT MITM, CR MITM, and MEP attacks. But the security provided by such hardware token based scheme depends upon the security provided by the issuer or manufacturer of that token. In 2011 RSA SecurID [64] tokens were compromised [78]. This caused replacement of 40 million SecurID tokens that were in use during that time. Also, it has been recently discovered and published that such hardware security keys can be reverse engineered and spoofed [66].
6. Password managers are an easy target for MEP attacks as the information that is auto-filled can be sniffed. Video in appendix describes hacking of auto-filled credentials by LastPass password manager in detail.

2.2.2. Usability and cost issues

1. OTP based schemes are popular, but they have an added cost per login which is a burden to organizations.
2. Graphical password-based schemes [20,44] are not easy to learn and use and the user may require extra training [55] for using them.
3. Some of the schemes that can resist these attacks are client dependent (Tricipher [68]). As schemes such as Tricipher rely on the trusted module on the client machine, the user can login only from the machine on which the trusted module is installed and on which the credentials are stored.
4. Separate hardware token-based authentication schemes such as RSA SecurID, Yubikey [79] require their users to buy and carry an extra security key or token. The additional cost and the need to carry a separate hardware device may be inconvenient to the users.

3. Proposed scheme

The proposed authentication scheme addresses some of the research gaps identified in the previous section. In the proposed scheme a user may login on a website either through his smartphone or through a PC (desktop/laptop). In the former case, only a smartphone is needed whereas, in case of latter, a smartphone, an Android App (or iPhone App) and a PC with a Chrome App installed on it are required. The proposed multifactor authentication scheme requires a user identification token, a user password (PWD) and an instance id of the Android/Chrome App.

Unlike other schemes where the user enters his login username manually on the system, the Android/Chrome App used in the proposed scheme fetches the Bluetooth address of the smartphone automatically and uses it as the user's login username. This not only increases the complexity of obtaining the Bluetooth address during a phishing attack but also makes it almost impossible for the attackers to obtain this information from users via reply to emails. Also, it is difficult to obtain, relay and use this information in real time to perform an RT MITM attack. As the Bluetooth [80] device remains physically paired with the user's desktop, it avoids the possibility of CR MITM attacks too. Currently we have only utilized the Bluetooth address of the smartphone for developing the proof of concept of this authentication scheme but other hardware information of the BT device such as device name, device class, vendor ID source, vendor ID, product ID, device ID, device type, UUIDs [81] can also be fetched from the device via the Bluetooth API and a hash (message digest) of the acquired information can be used as the username for better security. The user is required to remember and enter one user authentication token i.e. the password, on the system during the login process. App Instance Id (AID) which is used as a second authentication factor is procured automatically by the App. The scheme provides mobility and the user can use Chrome App to log in from any PC. While using a different PC the user should login over the Chrome browser of the PC to sync the Chrome App and its data. Once he/she logs out from the browser Chrome App gets disconnected. The use of Chrome App does not reduce the security of the user's Google account or compromise it in any way.

3.1. Assumptions and threats

3.1.1. Assumptions

1. PC is assumed to have an inbuilt BT adapter or an external plug-and-play BT adapter USB device, which is cheap and easily available. The smartphone is BT enabled and is in proximity to the PC.
2. Though an attacker can install spoofed apps (Android App and Chrome App) and extensions on a user's devices, during registration, it is assumed that the user uses the authentic Android App/Chrome App. Like many other schemes [21], it is assumed that the new user registration is free from attacks.
3. The proposal assumes that the data transfer between the client and the server over HTTPS is secure from sniffing and can be used to exchange secret keys. Authentic website servers and the information stored in their databases are assumed to be secure.
4. It is assumed that the organization provides a secure DNS service, and the attacks originating from a malformed DNS or session hijackings or through sophisticated host-based malware are currently considered outside the scope of this work**.** Also, the attacks due to source code modified browser installed on user's PC are out of the scope of this work**.**

3.1.2. Threats

We assign an attacker the following set of capabilities:

1. **Phishing, MITM Phishing:** The attacker can deceive users through a phishing website and can obtain user credentials. Additionally, an attacker can install remote desktop capturing/remote desktop relay modules (Teamviewer, Ulterius etc.) on the client machine or can use techniques such as QRLjacking to carry out CR MITM and RT MITM phishing attacks respectively.
2. **MEP Attacks:** Attacker can install a malicious browser extension and can obtain permissions needed for carrying out an attack by providing a novel functionality in the foreground that requires the same set of permissions. The malicious browser extensions installed on client's browser can log user keystrokes, browser screens and/or can sniff the information entered by the user on the website opened in a browser.
3. **App Spoofing:** Attacker can install spoofed Chrome App or Android App on the user's PC or smartphone and can deceive users into entering their credentials over Apps [82].

3.2. Abbreviations used

Variable	Description	Function	Description
U	Username	HTTPS $_{GET}$ (Parameters)	An HTTPS GET request with parameters.
PWD	Password	HTTPS (DATA)	HTTPS message carrying DATA.
EM	Email Address	REG $_{DB}$	Adds a new tuple (U, EM, PWD, PN, BT_{ADDR}, SALT and AID_{AAP}) to the server database.
PN	Phone Number	V $_{DB}$ (a, b, ...)	Matches user credentials received as arguments with those present in the server database.
AID_{AAP}	App instance id of the Android App	REPLACE (a, b)	Replaces the encrypted value 'a' with encrypted value 'b' in server database / Chrome storage.
AID_{CAP}	App instance id of the Chrome App	GENERATE (x)	Generate a cryptographically secure random secret x.
OTP	One Time Password	STORE (x)	Stores the value 'x', encrypted (symmetric encryption AES) with user PWD and SALT in Chrome App or Android App storage.
		ADD_{DB} (x)	Creates a new column in the server database and stores value 'x'.

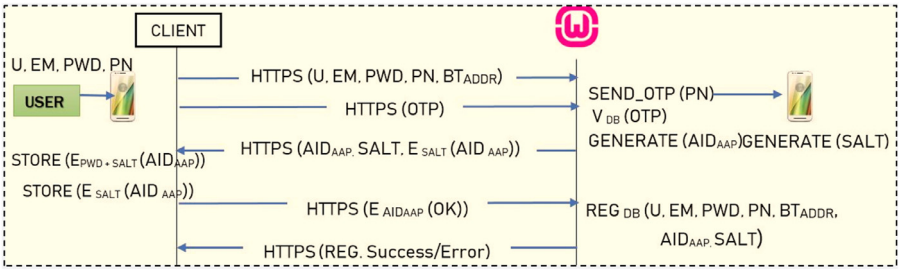

Fig. 3.1. Registration phase.

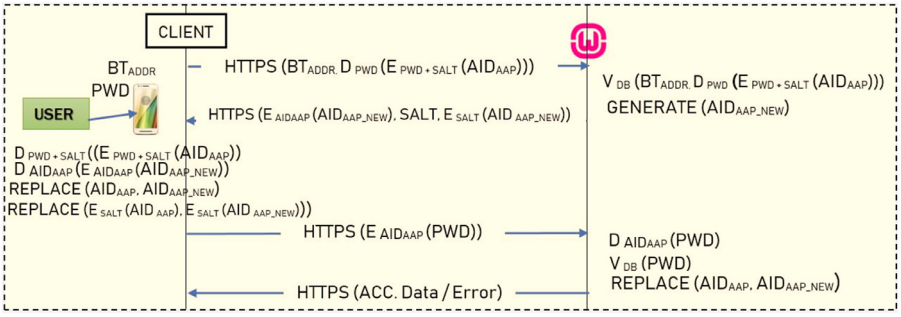

Fig. 3.2. Smartphone login phase.

3.3. Registration and login phases

A user can login to a website either using a smartphone or a PC. During the Registration Phase, a new user registers on the website for the very first time. A registered user can either log in using his smartphone - Smartphone Login Phase or using a desktop - Desktop Login Phase.

3.3.1. Registration phase

Fig. 3.1 illustrates registration phase.

1. In this phase, a new user registers his or her account with the website by entering the username (U), password (PWD), EM, and PN into the registration/sign up page in the website's Android APP (AAP) which the user has downloaded and installed from the App Store.
2. The BT_{ADDR} of the user's smartphone is obtained automatically by the Android App and is sent to the server along with the user's personal information (U, PWD, EM, and PN) over an HTTPS connection.
3. After receiving the user information, the server sends an OTP to the registered phone number of the user.
4. Once the OTP is verified by the server, an account for the user is created and a unique as well as secret, AID_{AAP}, SALT and the $E_{SALT}(AID_{AAP})$ to be used for desktop login session [83] are sent to the Android App over the HTTPS connection.
5. The Android App encrypts AID_{AAP} using a key generated by the concatenation of the PWD and the SALT and stores $E_{PWD+SALT}(AID_{AAP})$ in its local App storage. The App also stores the $E_{SALT}(AID_{AAP})$ in its local App storage.
6. The Android App sends back E_{AIDAAP} (OK) to confirm that the AID_{AAP} has been saved successfully in its storage. If the server receives a valid E_{AIDAAP} (OK) it sends a successful registration message, otherwise, it sends an error message.
7. PWD and SALT are not stored in the Android App. Registration phase happens only once per user.

3.3.2. Login phase (Smartphone login phase / desktop login phase)

Smartphone Login Phase: Fig. 3.2 describes the interactions during this phase.

1. In this phase, unlike conventional authentication schemes, the user only enters his account PWD into the Android App login screen opened on his registered smartphone.
2. Once the user clicks on the login button after entering his password, Android App initiates the communication with the server and provides the BT_{ADDR} of the smartphone and the $D_{PWD}(E_{PWD+SALT}(AID_{AAP}))$ for user identification.
3. The server identifies the specific user from the BT_{ADDR}. The $D_{PWD}(E_{PWD+SALT}(AID_{AAP}))$ received from the client is verified by the server to ensure that the HTTPS request for SALT is being initiated by the registered Android App. To verify the same the server encrypts the AID_{AAP} stored in its database with the user's PWD + SALT and then decrypts it with user PWD to generate a server copy of $D_{PWD}(E_{PWD+SALT}(AID_{AAP}))$.
4. If the $D_{PWD}(E_{PWD+SALT}(AID_{AAP}))$ received from the client and the $D_{PWD}(E_{PWD+SALT}(AID_{AAP}))$ computed by the server are same, server generates a new app instance id (AID_{APP_NEW}). The SALT, $E_{AIDAAP}(AID_{AAP\_NEW})$ and $E_{SALT}(AID_{AAP\_NEW})$ are sent to the Android App.
5. The Android App uses the SALT received from the server and the user PWD to decrypt the locally stored AID_{AAP}. After obtaining the AID_{AAP}, the AID_{AAP_NEW} is decrypted. The $E_{PWD+SALT}(AID_{AAP})$ in the App storage is then replaced with the $E_{PWD+SALT}(AID_{AAP\_NEW})$ to be used for the next login session. In a similar way, the $E_{SALT}(AID_{AAP})$ is replaced with the $E_{SALT}(AID_{AAP\_NEW})$ to be used for the next desktop login session.
6. The Android App then uses the AID_{AAP} of the current login session to generate $E_{AIDAAP}(PWD)$ and sends it to the server for user verification.
7. The server decrypts the PWD with the old AID_{AAP} stored in its database for the BT_{ADDR} of the user.
8. If the $D_{AIDAAP}(PWD)$ matches with the user's PWD stored in the server database, the login attempt is successful, and the user account data is returned over the HTTPS connection.
9. Finally, the server replaces AID_{AAP} stored in its database with AID_{AAP_NEW}.

Desktop Login Phase:

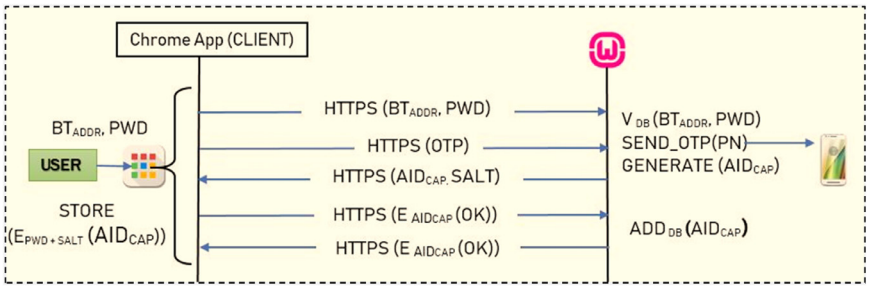

Fig. 3.3. Chrome App initial registration.

The prerequisite for login using a PC is to download, install and register the Chrome App.

- **Chrome App Registration:** Downloading and installing the Chrome App on the browser and its registration on the server is a one-time process and should be done after registration phase.

1. User downloads installs and opens Chrome App (CAP). A registration screen appears on the screen. The user connects his registered BT smartphone with the PC. Chrome App sends the BT_{ADDR} of the connected smartphone, along with the PWD entered by the user on the registration screen, to the server for verification.
2. Server, after verifying BT_{ADDR} and PWD, generates an OTP and sends it to the user's phone number. The user enters this OTP on the Chrome App which sends the OTP to the server.
3. After verifying OTP, the server generates a unique secret $AID_{CAP,}$ and sends it along with the user's SALT. Chrome App stores the received AID_{CAP} encrypted with user PWD and the SALT i.e. $E_{PWD+SALT}$ (AID_{CAP})) in its secure local Chrome App storage. Chrome App sends back a predefined message E_{AIDCAP} (OK).
4. The server, after verifying the received message adds the AID_{CAP} in the user record and sends a registration success E_{AIDCAP} (OK) or error message.
5. Chrome App registration is done once per user after which users can sync the registered Chrome App and its data (including the AID) to any PC at any location by logging into their Chrome browser (Fig. 3.3).

- **Desktop Login**

For login, the user first connects his BT registered smartphone with the PC.

1. The user connects his Smartphone via BT with his PC. The user then opens the Smartphone App and click on "Desktop Login" button. The App fetches the E_{SALT} (AID_{AAP}) stored in the App storage and writes it on the BT_{NAME} property of the Smartphone's BT Adapter. The name property also gets reset to its original value after 60 seconds by the Smartphone App. We have currently used the BT_{NAME} property for our testing but any miscellaneous BT property of the device can be used for communication of the E_{SALT} (AID_{AAP}) value. The reset time can also vary in a real implementation based or user experience.
2. The user then opens the Chrome App on the browser which displays a login screen for the user entry of PWD. After entering his account PWD the user clicks the submit/login button and the Chrome App fetches the BT_{ADDR} and BT_{NAME} properties of the connected BT Smartphone device and send the BT_{ADDR} of the smartphone along with the $D_{BT\ NAME}$ ($E_{PWD+SALT}$ (AID_{CAP})). Where BT_{NAME} contains E_{SALT} (AID_{AAP}).
3. The server identifies the user through BT_{ADDR} and $D_{BT\ NAME}$ ($E_{PWD+SALT}$ (AID_{CAP})) is computed and verified at server's end to ensure that the request for SALT is coming from an authentic user.
4. After user verification, the server sends the AID_{CAP_NEW} (to be used for the next login session) encrypted with AID_{CAP} and the SALT corresponding to the user account in the response with a session token TK (valid only once and for a short duration to login over a specific user account) encrypted with the AID_{CAP} of the ongoing session.
5. Chrome App uses the user's PWD and the received SALT to decrypt the $E_{PWD+SALT}$ (AID_{CAP})) stored in its local storage. The AID_{CAP} retrieved is then used to decrypt the E_{AIDCAP} (AID_{CAP_NEW}) to retrieve the AID_{CAP_NEW} to be used for the next login session. Chrome App also replaces the storage with the newly received value. Fig. 3.4.
6. Chrome App decrypts the E_{AIDCAP} (TK) and creates an HTTPS GET URL request to the authentic website (AW) URL with E_{AIDCAP} (PWD, BT_{ADDR}) and the TK as the parameters. The parameters can also be sent as a POST request.
7. The Chrome App initiates a command to open a new tab on the Chrome browser and the tab URL is updated with the URL created in step 6. The role of the Chrome App to carry out a secure user authentication with the server ends here and now the server receives the parameters and the TK in an HTTPS request directly from the browser itself.
8. The TK is extracted and if it is valid and corresponds to a user account, then the associated AID_{CAP} is used to decrypt the GET parameters received. The extracted BT_{ADDR} and PWD are matched with the user account. If matched successfully results in a successful login and the user's account data is returned in response to the browser.

3.4. AID and SALT generation

The authenticators AID_{CAP}, AID_{AAP} and the SALT are generated using AES in Counter (CTR) mode. AES in CTR mode can be used as a cryptographically secure pseudo-random number generator (CPRNG). An initial 16-character random string is generated by choosing an ASCII character for individual character positions using random function. The random function generates a random number (0–255) using the current server timestamp as the seed value. Server timestamp (TS) in specific corresponds to the local timestamp of the machine when NTP servers are synchronized to time.google.com.

$$srand(servertimestamp);$$

$$CharID = rand(0,\ 255)$$

The random number generated each time (CharID) is used as an index to choose the specific character from the ASCII set which gets stored at i^{th} position in the Key. The process is repeated 16 times. The resulting 16-character random string is used as the key for AES CTR mode block cipher encryption with an initial counter

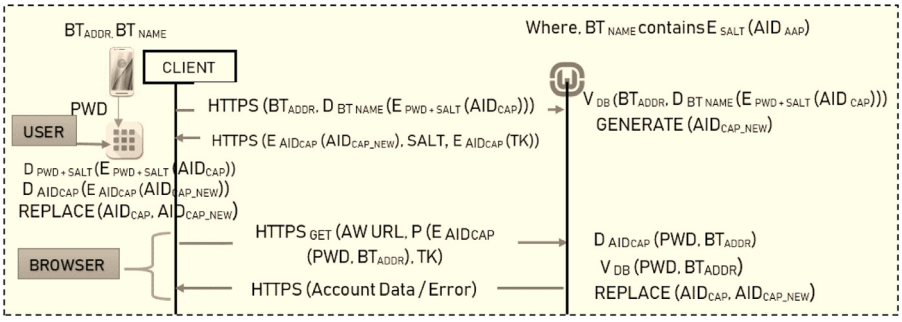

Fig. 3.4. Desktop login phase.

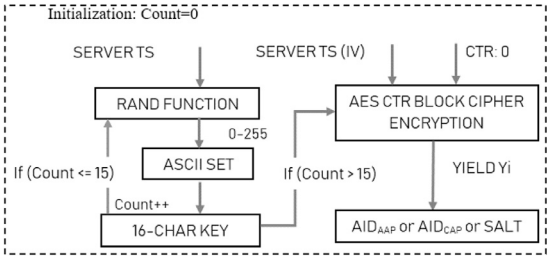

Fig. 3.5. Generating AID_{AAP}, AID_{CAP}, SALT.

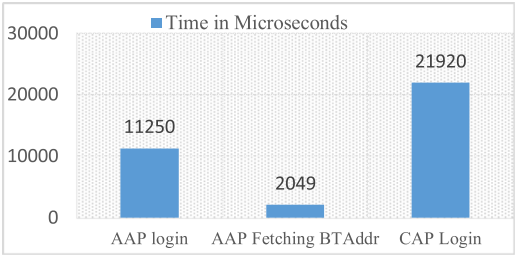

Fig. 4.1. Time needs through AAP and CAP.

value of 0 and Initialization Vector (IV) (nonce) as the server current timestamp. The resulting 16-character yield Yi is then used as the AID_{AAP} or AID_{CAP} or the SALT as described in Fig. 3.5.

$$Yi = F(Key,\ IV + g(i))\ Where,\ g(i) is\ any\ deterministic\ function$$

The secret keys generated in this way are more cryptographically secure than those generated using a raw random key generator [84]. The generated AID is utilized for symmetric encryption of HTTPS GET parameters and other message exchanges using AES encryption.

4. Implementation, testing and security analysis

4.1. Test setup

The following hardware and software were used for the implementation and testing.

- A Motorola E3 smartphone with MTK6735 Chipset, 1.0 GHz Quad Cortex A53 CPU, 2GB RAM, and Android Marshmallow operating system.
- The PC used for desktop login phase was a Windows 10 machine with Chrome Browser 58 Version and an Intel® Core™ i3-5020 2.20 GHz CPU with 6 GB of RAM.
- The website for testing registration phase, Smartphone login phase, and Desktop login phase was hosted on a local PC running Windows 7 OS on an Intel® Core™ i7-3770 CPU having 8 GB of RAM. The website was written in PHP and MYSQL and was hosted on WAMP server version 2.5.
- Chrome.Bluetooth API was used to develop the Chrome App. The PC hosting the website, the PC used for login and the smartphone were on the same LAN during the testing.
- Though the Chrome App and Android App have been used for implementation of the demo, the proposal can be implemented in other browsers and mobile operating systems.

4.2. Performance evaluation

We evaluated the performance of our proposed scheme in terms of time taken for its various operations and the CPU and

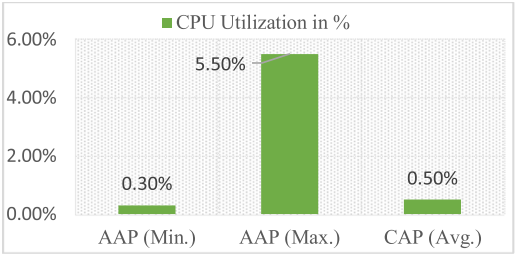

Fig. 4.2. CPU utilization: AAP & CAP.

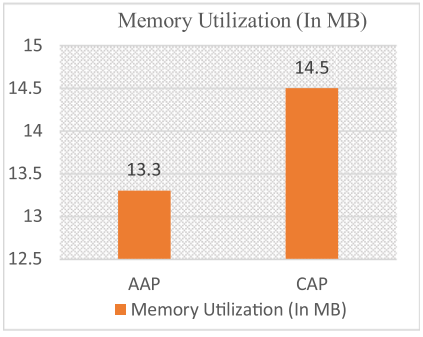

Fig. 4.3. Memory utilization: AAP & CAP.

Memory utilization. "*Another Monitor*" Android App [85] was installed over the smartphone to record the CPU and Memory utilization (Fig. 4.5) of the Android App (named Authenticate) while CPU and Memory utilization of Chrome App was recorded using the inbuilt Chrome browser task manager (Fig. 4.4.). It can be seen from Fig. 4.5 that the minimum CPU utilization of AAP was recorded to be 0.3% (rightmost snapshot), the maximum CPU utilization was 5.5% and the average memory utilization was 13,332 kB (leftmost snapshot). The statistics confirm that the CPU and Memory utilization for our scheme is low enough and it can be utilized for day-to-day login. Figs. 4.1, 4.2 and 4.3 show the recorded values.

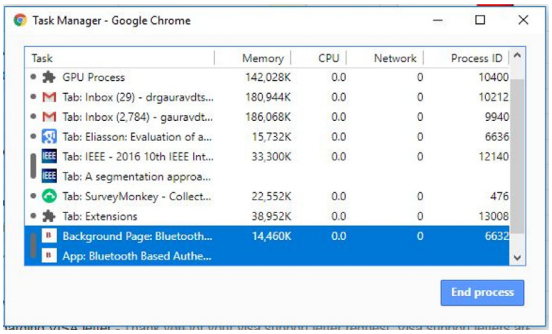

Fig. 4.4. Statistics via Chrome task manager.

The login time and CPU and memory utilization of our scheme were not compared with other authentication schemes because of the following reasons:

- Login time depends upon the availability and speed of the application server. The login time shown in Fig. 4.1 was recorded when a single user attempts to login to our server. It was not possible to reproduce on a commercial website following other authentication schemes (U-PWD, QR Code based, push notification based login) because commercial websites serve multiple customers at the same time.
- CPU utilization and memory utilization may vary based on the kind of application offered by the App. For example, an online social networking app such as Facebook App consumes high CPU and Memory because of heavy media content it receives, tracks or process. Hence directly comparing Apps CPU Utilization is not a good idea.
- Other non-commercial schemes proposed in the past does not mention login time, CPU and memory statistics.

We also did experiments to record the average additional time it takes for a smartphone (that has been paired with a PC at least once in the past) to get connected via Bluetooth when a Bluetooth connection is initiated between the Smartphone and the PC. An average of 100 experiments is shown in Fig. 4.6. The values for graphical password entry time and additional time of receiving push notification has been obtained from [86] and [53] respectively. Results show that time needed to connect a smartphone with a PC via Bluetooth is much smaller than the time it takes on an average to receive an OTP or push notification, or scan a QR code or enter a graphical password / CAPTCHA.

4.3. Security analysis

4.3.1. Security against known threats

Table 4.1 discusses the security of the proposed scheme against the attacks which can be carried out by an attacker to compromise the authentication scheme or steal the user credentials.

One must also note that the scheme will not get compromised even when the attacker will spoof the user's registered Smartphone's Bluetooth address on his/her Bluetooth device via tools such as bdaddr [89]. This is because in such a case the attacker will be able to provide the BT_{ADDR} corresponding to the user account for authentication but won't be able to provide a valid E_{SALT} (AID_{AAP}) corresponding to the user account which is only known to the authentic Android App installed on the user Smartphone.

4.3.2. Comparison: security against known threats

This section compares the security provided by the existing schemes (Table 4.2) and our proposed scheme in terms of security against the attacks described earlier in Section 3.1.2.[1]

- RT MITM and CR MITM Phishing Attacks
- ***OTP/PIN-based schemes:***
- Google 2 step [49] OTP based authentication and SAASPASS [48] authentication via secret key can be easily broken using a phishing website. The Username-Password (U-PWD) and the OTP/Secret key entered on the phishing website can be relayed in real time to the authentic website [See Appendix for details].

[1] The attacks can be reproduced following the steps given in the document mentioned in the Appendix.

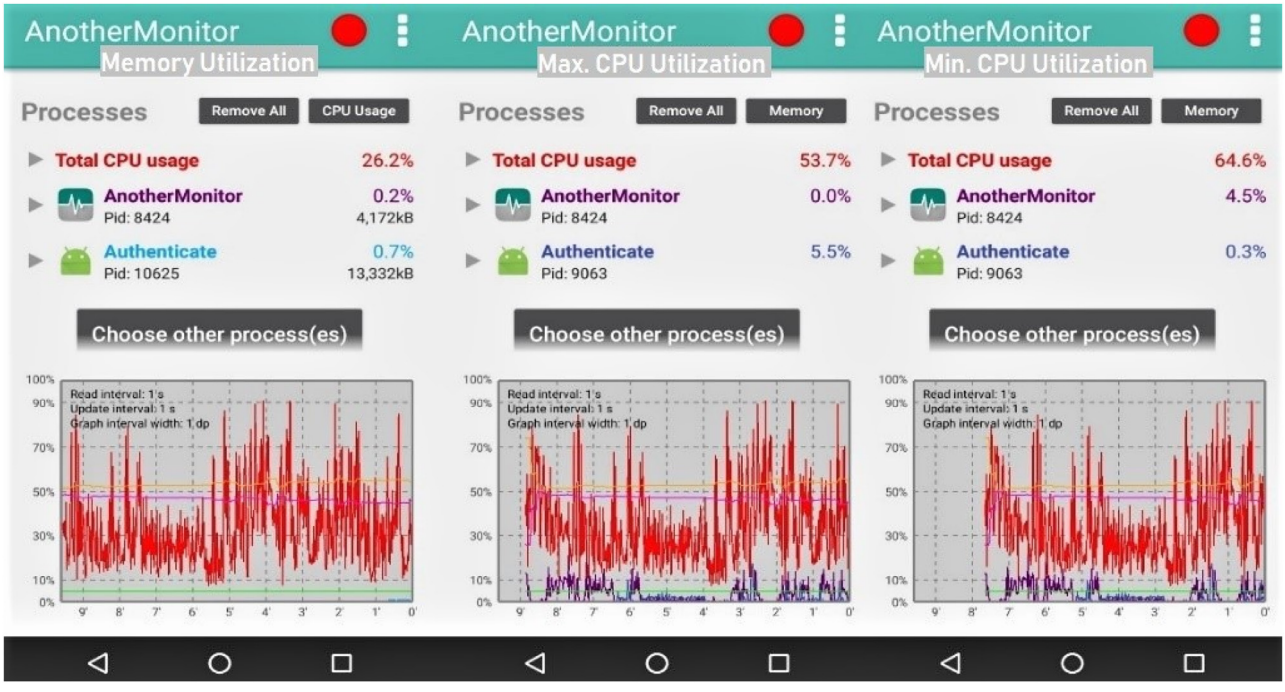

Fig. 4.5. Memory and CPU utilization stats of Android App (Authenticate) on a smartphone.

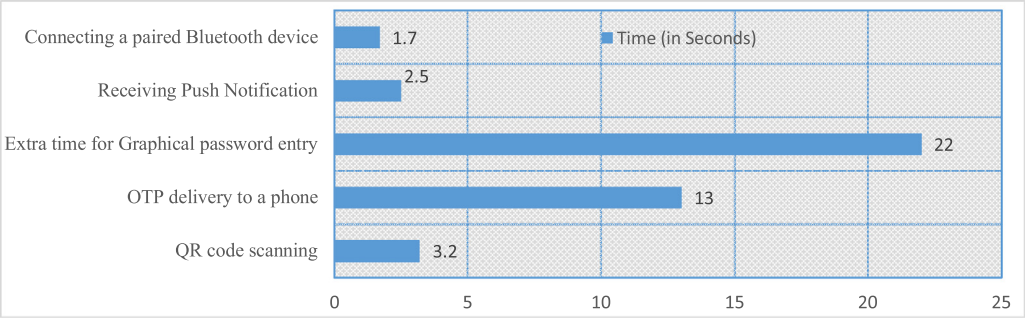

Fig. 4.6. Comparison of time needed for carrying out additional operations during authentication.

Table 4.1
Security analysis against known threats.

Threat	Security Justification
Traditional Phishing	Bluetooth address and AID_{CAP} are automatically sent via the Chrome App to the authentic website. The password is also entered by the user on the Chrome App. Traditional phishing attacks requesting user information over phishing websites cannot be carried out as a phisher cannot obtain all the credentials from the user on a phishing website.
RT / CR MITM	Bluetooth address and user-specific AID_{CAP} cannot be acquired through remote desktop monitoring / remote screen relaying, malicious browser extensions or on phishing websites and hence cannot be relayed to an authentic website in real time to cause an RT MITM attack or a CR MITM phishing attack. Even if Bluetooth address and password are acquired through a spoofed Chrome App or phishing website, the attacker will not be able to obtain the AID_{CAP} which is stored encrypted (encryption key: user password concatenated with server SALT) in the local storage of the authentic Chrome App due to same origin policy. Also, the attacker will need the E_{SALT} (AID_{AAP}) stored on the registered smartphone Android App.
MEP Attacks	Chrome Apps, unlike websites, runs on a dynamically allocated localhost URL after their installation. Malicious browser extensions or peer apps cannot access the information (via sniffing, keylogging) that is entered or stored (user password or AID_{CAP}) on a Chrome App due to same origin and the strong isolation properties provided by the browser platform. Also, the AID_{CAP} is stored in an encrypted form using multiple credentials where one part is the user PWD which is concatenated with the other part i.e. the SALT stored at the server [87,88].
App Spoofing	Spoofed apps installed on a smartphone or a PC can steal user credentials, such as user password and BT_{ADDR}, but will not be able to acquire AID_{CAP} or AID_{AAP}. Similar to the Chrome App the smartphone App also runs as a separate process and hence due to process isolation property the Encrypted AID_{AAP} is also not accessible to peer malicious spoofed Apps installed on the mobile device.

Table 4.2
Comparison with existing proposals in terms of security against known threats.

SCHEME	MITM		MEP attack			AS	*Reasonably Secure Count*
	CR	RT	K	S	P		
1. Google 2 Step [49]	✗	✗	✗	•	✗	✗	***0***
2. SAASPASS [48]	✗	✗	✗	•	✗	✗	***0***
3. Xie et al. [21]	√	✗	√	•	√	✗	***3***
4. Kim et al. [19]	✗	✗	√	√	√	✗	***3***
5. Mukhopadhyay et al. [71]	✗	✗	•	•	•	√	***1***
6. Dodson et al. [72]	✗	✗	√	√	√	√	***4***
7. Leung et al. [20]	✗	√	√	•	√	•	***3***
8. Zhu et al. [44]	✗	✗	•	✗	•	√	***1***
9. Tricipher [68]	√	√	•	•	•	√	***3***
10. RSA SecurID HW token [64]	✗	✗	✗	•	✗	✗	***0***
11. Yubikey U2F [63,79]	√	√	√	•	•	√	***4***
12. Push Login [54,90]	✗	✗	•	•	•	✗	***0***
13. Password Managers [51,91]	√	√	✗	•	✗	√	***3***
14. U-PWD [92]	✗	✗	✗	•	✗	✗	***0***
15. Proposed Scheme	√	√	√	•	√	√	***5***

Note: √→ Reasonably Secure, ✗→ Insecure and the scheme will fail, •→ Partial information can be accessed but the scheme will not be compromised. Reasonable secure count→ shows the count of the number of ticks and hence an indicative measure of the level of security against known threats.

Even if the scheme checks the proximity using a BT device, the attacker can acquire and spoof the BT_{ADDR} of the user's device before performing the RT MITM attack. The schemes are vulnerable to CR MITM attacks as all the credentials can be obtained by an attacker on his remote desktop session through deception.

- ***QR Code scanning based schemes:***
- The scheme proposed by Xie et al. [21] can be compromised by the attacker using a spoofed malicious browser extension. The malicious extension can obtain the username and password. Then the attacker can enter this information on the CamAuth extension installed on the attacker's browser. The CamAuth extension initiates the Diffie-Hellman key exchange and sends the user information to the server. After user verification, the server sends the barcode to the attacker's browser. Attacker displays it on the spoofed malicious browser extension installed on the victim's machine. The victim's CamAuth smartphone App scans the Barcode and decrypts the information using the server's public key stored on it and creates a vouch which is displayed in the form of a Barcode and scanned by the PC camera. The attacker obtains the Barcode from the malicious browser extension and copies it as an image on his/her smartphone. The authentic website opened on the attacker's computer obtains the Barcode image using the PC camera. After obtaining the Bar-

code, the server allows an attacker to log in. CR MITM is not possible because the attacker's PC cannot access the PC camera of the victim's machine to scan the Barcode generated by the smartphone App.

- The QR code generated in the scheme proposed by Kim et al. [19] contains the IP address of the user's computer and a session id. The browser on the attacker's machine can send a login request to the authentic website with the spoofed IP address of the targeted victim's PC. The authentic website will generate a QR code and will send it back to the attacker. An attacker can easily relay this QR code to the phishing website opened by the victim through techniques such as QRljacking. The victim will use his/her authentic QRA App to scan the QR code. The QRA App will verify the IP address based Geolocation. As the attacker has spoofed his IP address, the QRA App with user credentials will verify the session between the attacker's browser and server. The server, after getting the verification token for the session from the QRA App, allows the attacker to log in. Both CR and RT can break this scheme.
- In Mukhopadhyay et al.'s [71] study, U-PWD can be accessed via a phishing website and can be sent to the authentic website (Identity Provider) IDP. A QR code challenge generated by the IDP (encrypted by the user secret key X_A) can be relayed to the phishing website. A user under deception will scan the QR code from his/her smartphone and will decrypt the challenge using his/her secret key X_A stored on the user smartphone. The smartphone will send the response to the IDP which will provide its assertion to the authentic website, allowing the attacker to log in to the authentic website.
- In the case of Dodson et al. [72], all the secret information is stored on the user's smartphone which is needed to verify a user's desktop browsing session. To break the scheme, the attacker can obtain the QR code of his browser session from the authentic website and relay it to the phishing website opened by the user. The user under deception scan the QR code, eventually verifying the browser session (attacker's) challenge given in the QR code (Appendix showcase a practical attack scenario on WhatsApp web).
- ***Separate hardware token based schemes:***
- Tricipher [68] is safe from RT, CR MITM as multipart credentials are used. Not all information can be entered, scanned, or logged by any phishing website.
- Security keys such as RSA SecurID soft token / hardware token can be compromised through RT MITM phishing by obtaining the RSA passcode on a phishing website or through a malicious browser extension and then using it in real time on authentic website (within 30 seconds) to compromise the scheme (see example of SAASPASS in Appendix). Yubikey using U2F protocol is however reasonably safe from RT MITM and CR MITM phishing attacks.
- *Graphical password:*
- The scheme proposed by Leung et al. [20] is safe from RT MITM as the user mouse click coordinates used for entry of user password on a moving CAPTCHA are difficult to relay. Screen resolutions of attacker's PC must also match with the victim's PC which also increases some level of complexity. CR MITM is however possible.
- Zhu et al.'s [44] process can also be compromised by RT MITM since the graphical password obtained from the authentic website can be relayed to the user on a phishing website and the coordinates of the mouse click can be recorded. The recorded coordinates can be mapped by the attacker to his screen and the corresponding points can be clicked on the CAPTCHA shown on the authentic website opened on the attacker's machine. CR MITM can also take place.
- ***Push notification based login:***
- Push login [90] fails to handle the RT MITM and CR MITM phishing attacks. An attacker can receive the username (password etc.) on a phishing website and simultaneously can enter it on the authentic website. A user can also enter the username (password etc.) under deception on an attacker's browser (CR MITM) displayed on his desktop. Subsequently, in both the cases, the victim user under deception will approve the push notification obtained on the smartphone App [See Appendix for more details of this attack].
- ***Password managers:***
- Password managers [51,74] safeguard the user from RT MITM and CR MITM, as no credentials are entered by the user on the website. Therefore phisher cannot get and relay any credential even if the victim opens the phishing website. This is based on the assumption that the user waits for the password manager to enter the credentials for the websites registered with the password manager and he will find it suspicious when the password manager won't enter the credentials on the phishing website (being not on the list of websites whose username and password is stored with it). However, an attacker can still deceive the user into re-entering the credentials by entering random credentials by itself and once the user enters the submit button it can alert the user about wrong credentials. In such a case user may re-enter the credentials leading to RT MITM and CR MITM.
- The basic U-PWD schemes are vulnerable as the information entered can be relayed from phishing website to authentic website. The proposed design reduces the possibility of RT MITM and CR MITM to almost zero as nothing is entered on a website which can be accessed via a phishing website and relayed to an authentic website. There are multiple credentials used during the desktop authentication and each known to separate entities which include the user (PWD), PC (AID_{CAP}) and Smartphone (E_{SALT} (AID_{AAP})). Also, all the information during the authentication is sent from the user-specific Chrome App. Spoofing of Chrome App and its effects will be discussed later in this section.
- **MEP Attacks**
- In the case of MEP attack, an attacker tries to steal user credentials by key logging (K), screen logging (S), or password sniffing (P). If this can cause a loss of all credentials that are needed for authentication during login, the scheme is considered vulnerable to MEP attack.
- Keylogging and password sniffing can break U-PWD, Google 2 Step [49] OTP authentication, SAASPASS [48] PIN-based authentication, RSA SecurID hardware / soft tokens and password manager based [51] authentication, because in these schemes a user and/or the hardware enters the complete set of credentials on the website, including the second factor and which can be sniffed via a malicious browser extension. In case of password manager [51], the auto-filled credentials can be sniffed by the malicious browser extension installed on the client before they are transferred to the server. Appendix [Appendix] shows a video demonstration of the same.
- In Xie et al. [21], the information is entered on the CamAuth extension so malicious browser extensions cannot access the information entered by the user using key logging or password sniffing, due to the same origin policy. A QR code can be accessed via screen logging but this is not enough to break the authentication.
- Similar reasoning applies for Leung et al.'s [20] method, where the flash-based moving OTP CAPTCHA is shown in the browser extension which makes keylogging or password sniffing impossible. However, in Zhu et al [44] approach a malicious browser extension can use screen logging to capture the user input on the static CAPTCHA password to obtain the user password.

- Kim et al. [19] and Dodson et al.'s [72] methods are resistant to MEP attack as the user information is never entered on the website.
- In Tricipher et al. [68], Mukhopadhyay *et al* [71], and Zhu *et al* [44], and Push login [90], malicious browser extensions can use key logging and/or password sniffing to acquire the identification or authentication tokens of a user but still these schemes cannot be compromised due to the involvement of a trusted device or second authentication factor, which cannot be accessed via malicious browser extensions.
- The proposed scheme is secure against MEP attack as keylogging and password sniffing cannot be performed by a malicious browser extension due to same origin policy. Also, the password is entered as dotted characters and cannot be obtained via screen logging.
- App Spoofing (AS)

An authentication scheme can be compromised using App spoofing if spoofed Android Apps/Extensions can be used to steal authentication credentials. Xie et al. [21], Kim et al. [19], Google 2 Step [49], SAASPAAS [48], U-PWD and Push Login [90] are vulnerable to this type of attack. A spoofed Android App installed covertly on a user's smartphone can obtain all credentials needed for user authentication as all these credentials are entered on the App at the time of login. Schemes such as Tricipher [68], Mukhopadhyay et al. [71], Dodson et al. [72], password managers [51,74], and the proposed scheme are secure from such attacks as in all these schemes at least one credential is stored on the user device and is not entered on the App at the time of login. The schemes like Zhu et al.'s [44] which do not utilize any Apps or Extensions are safe from these attacks.

4.3.3. Comparison: usability

This section compares authentication schemes in terms of the number of authentication tokens used by the scheme, the number of authentication tokens to be memorized**,** additional needs and the need of internet on phone for the scheme to work for desktop login**.** Most of the existing schemes only require a smartphone as an additional need which the Internet users mostly carry. Others require PC camera, specific driver modules installed on the PC, GPS, hardware token, or trusted third party etc. Few schemes [20,51] do not require any additional resources for their operations. The need for a WIFI or Cellular data connection on the smartphone while logging in from a PC is an important factor to consider for understanding both the cost incurred when using the scheme and the issues that can arise due to a lack of Internet connectivity on a smartphone. Table 4.4 describes the comparative discussion.

4.3.4. User survey

To understand user perception of the authentication schemes we conducted a pilot user survey. Respondents were selected from different background, age group, computer proficiency to avoid the possibility of a convenience sample. The goal was to get user feedback on the security and user-friendliness of our scheme in comparison to other authentication schemes. Feedback from 194 users was collected for the seven authentication schemes: (1) U-PWD, (2) U-PWD and OTP, (3) QR Code based, (4) Graphical PWD based, (5) Hardware token / RFID card based, (6) Proposed Scheme, and (7) Push login notification based schemes. Table 4.5 shows the age group and computer proficiency of the participants.

Readers should note that a very few people having an age of 40 years and above participated in the survey and hence the survey does not reflect the opinion of elderly aged people. We were not able to attract a lot of people from this age range to fill the online survey and we plan to do an offline paper survey to collect their responses in future. Also, most of the people who appeared in the survey were either computer scientist/engineer or the people who use a computer frequently. Though we acquired responses from probable users of the scheme, we are also planning to cover more people who use computer infrequently to know their opinions via an offline survey in future. The survey was administered online. A brief introduction of all the schemes was given to the users who are either computer scientist/engineer or use computers frequently before the survey. However, a detailed introduction and hands-on of the schemes was given to the users who use computers infrequently. After the introduction to the schemes, users were asked to do the following:

1. Rate individual schemes from 1 to 5 (5 - Very Easy, 1- Not Easy) with respect to "Easy to Learn and Use".
2. Specify scheme(s) that are secure as per their knowledge.
3. Specify scheme(s) that they will prefer to login over a website containing their important data (e. g. banks, financial accounts, etc.)
4. Choose scheme(s) that should be implemented on all websites in the future because they have a balance between "Easy to Learn & Use" and "Security".

In the case of questions 2, 3 and 4, users were allowed to select multiple schemes. The results of the survey are given in Fig. 4.7.

1. Quantitative Analysis:

- For question 1, the numbers above the bars show the average score (on a scale of 100) given by the users to each scheme. Users rated U-PWD as the easiest scheme. OTP based two-factor schemes placed second. Our proposal and QR code based schemes almost got the same scores and were voted at third and fourth place respectively. Graphical password-based schemes, push notification based login schemes had nearly the same scores. Hardware token-based authentication schemes scored the least.
- 55 out of 194 users voted for the proposed scheme in the case of question 2, which was substantially more than the number of votes received by U-PWD, QR Code based, or graphical password based schemes and almost similar to the votes received by the push notification based login schemes and separate hardware token based schemes. This shows that users were more confident about the security of our proposed schemes in handling different attacks. The highest votes were received by OTP based 2FA schemes.
- Votes received in answer to question 3 also indicate that the users prefer the proposed scheme more than the hardware token-based authentication scheme for logging in to important web accounts (such as banks etc.) and almost equivalent to push notification based login schemes. Users showed more confidence in the proposed scheme compared to U-PWD, QR Code, and graphical password-based schemes. The highest confidence of users was however received by OTP based 2FA schemes.
- From the votes received in response to question 4, it is evident that the users want to use the proposed scheme over other authentication schemes except for the OTPs for login over websites in future.

Though OTP and push notification based schemes received higher or almost equivalent votes compared to the proposed scheme, they do not provide the same level of security against known threats as provided by our proposal (discussed in Table 4.3).

2. Conversation with survey respondents

After the survey, we interacted with 105 respondents who were available for conversations to understand the reasons for their opinions. Conversations revealed the basis of the scores.

Table 4.3
Comparison in terms of number of tokens used and their security.

Scheme	Tokens used by the scheme	Tokens to be remembered by the users	Additional Needs	The need of Internet on Phone
1. Google 2 Step [49]	3 - U, PWD, OTP on SP	2- U, PWD	Cellphone	N
2. SAASPASS [48]	3- U, PWD, OTP on App	2- U, PWD	Smartphone	Y
3. Xie et al. [21]	4 – U, PWD, DH Public. (g, p), Private U p	2 - U, PWD	PC Cam, Smartphone	N
4. Kim et al. [19]	4- U, PWD, Session ID, Secret key	2- U, PWD	Smartphone with GPS	Y
5. Mukhopadhyay et al. [71]	3- U, PWD, Secret key in SP	2- U, PWD	Smartphone, TTP	Y
6. Dodson et al. [72]	4 – U, PWD, Secret Key, QR Code	0 - NIL (QR Code Scan)	Smartphone	Y
7. Leung et al. [20]	4 – U, PWD, Secret Key, OTP CAPTCHA	2- U, PWD	NIL	NA
8. Zhu et al. [44]	3 - U, SALT, PWD CAPTCHA)	2- U, PWD	NIL	NA
9. Tricipher [68]	4 – U, PWD, TPM Secret Key, TACS credential	2 – U, PWD	CAPI Driver, Separate Hardware, TPM	N
10. RSA SecurID HW token	4 - U, PWD, HW token information, PIN	2 – U, PWD	Separate Hardware	NA
11. Yubikey U2F	5 – K_{PUB}, K_{PRIV}, Counter, U, PWD	2- U, PWD	Separate Hardware	NA
12. Push Login [90]	3 – U, PWD, SP	1 - U	Smartphone	Y
13. Password Managers [51]	3 – U, PWD, the master key of the password manager	Master PWD	NIL	NA
14. U-PWD [92,93]	2 – U, PWD	2 - U, PWD	NIL	NA
15. Proposed Scheme	4 – BT_{ADDR}, PWD, AID_{CAP}, AID_{AAP}	1 - PWD	Smartphone	N

Note: TPM → Trusted Platform Module, U→ Username, PWD→ Password

Table 4.4
Surveyed users' information.

Age Group (Year)	Participants	Computer Proficiency	Participants
15–25	110	Computer Scientist / Engineer	136
25–40	77	People who use Computers Frequently	45
40–60 and above	6 and 1	People who use Computers Infrequently	13

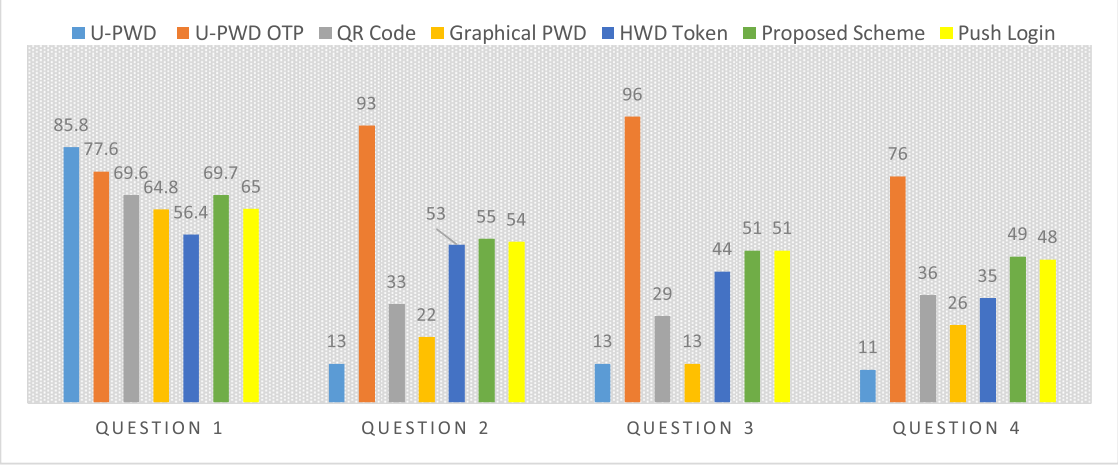

Fig. 4.7. User Survey Statistics.

- We concluded that users gave low scores to our proposed scheme in question 1 because they have never used the proposed scheme in the past. We also noted that the users were unaware of login via Bluetooth devices and they were interested in knowing more about our proposed scheme.
- The OTP and push notification based login schemes were voted better than the other schemes because users were more familiar with these schemes. From the conversations, we concluded that as most of the secure web logins (over banks, financial companies, and online shopping sites) request the user to enter an OTP during a web login; the user community has a perception that a secure login always happens when there is a need to enter an OTP.
- Also, around 60 percent of the users of all types of computer proficiencies were found unaware of the security vulnerabilities in the existing schemes that they use every day.
- People voted more for push notification based login schemes as the schemes are available in the market and approval of push notification through their smartphone app provides them the confidence of being secure.
- Most of the users (80%) told that graphical password based schemes are hard to use.
- Even when hardware token based schemes are more secure and can be used by the users for login to important web accounts, around 85% of the respondents did not appreciate the use of hardware tokens for login over websites. The cost of the separate tokens and the need to carry them always were the reasons reported by the respondents.

From the above survey, we concluded that our proposed scheme is more usable compared to separate hardware token based schemes that provide almost the same level of security. Also, the scheme is more secure than the schemes that provide a same or high level of usability (OTP, push notification based login). The proposal hence provides a balance between security and usability. For more information, the survey can be accessed directly at https://www.surveymonkey.com/r/2JCTQSR.

4.3.5. Comparison using web authentication assessment framework by bonneau et al. [92]

This section compares the existing schemes in terms of usability, deployability, and security using the Bonneau et al. framework. Some of the values shown in the comparison table have been taken directly from the study [92] and some of the values have been modified based on our analysis of the schemes in the light of newly identified cyber-attacks. To limit our discussions we provide only a brief description of various Bonneau et al.'s framework parameters and discussion of various schemes that offer the benefit provided by the specific framework parameter wherever necessary.

1. ***Memory wise effortless:*** Most of the schemes are not memory wise effortless (represented by✗) as they require users to remember their username and password. Kim et al. [19], Yahoo Push login [90], password managers [51,74], and our proposed scheme reduce the number of tokens to be remembered and entered by the user and hence are considered partially offering the benefit (represented by ❍) while Dodson et al.'s method [72] does not need any credential to be entered (represented by ✓).
2. ***Scalable for users:*** An authentication scheme is not scalable for users (from a user perspective) if login over multiple accounts using the same scheme increases the burden on the user. For example, receiving and entering OTP or PIN for logging in to each website is cumbersome. Our proposed scheme offers this benefit as in the future (discussed in the future work in section 5.2), the user will just need to connect the BT device with the PC once, and he/she can then log in to different websites by just changing the user password. Even the current version can be used for login to different websites (each providing their own Chrome Apps) with a considerable ease.
3. ***Nothing and quasi-nothing to carry:*** We consider that carrying a smartphone is nearly equivalent to nothing to carry and consider it in the category of quasi-nothing to carry given the fact that most people regularly carry a smartphone. If the user is required to carry a specific device or hardware, then the scheme is dependent on an extra device for authentication. Tricipher [68] does not offer this benefit as login is client system dependent.
4. ***Physically effortless:*** If the scheme does not require extra physical efforts beyond entering characters or pressing a button, then it is considered as physically Effortless. Most of the schemes do not completely offer this benefit as a user may need to interact with his smartphone to obtain OTP or PIN. Others which need more physical interaction such as QR Code scan, CAPTCHA entry are not physically effortless. Our proposed scheme is reasonably physically effortless as the Bluetooth pairing is a one clicks process.
5. ***Easy to learn:*** If an average user with basic computer knowledge can understand and use the scheme, then the scheme is considered easy to learn. Most of the schemes which use scanning of QR Codes, CAPTCHA, or graphical passwords do not necessarily offer this benefit. The proposed scheme is, however, easy to learn as the user only needs to learn how to perform the BT pairing. Also, BT device once paired with a computer can be connected to the same computer by just one click. So the old age people or people lacking technical knowledge can be helped during the BT pairing (exchange of secret passcode etc.) of their Smartphone with the PC and after which they can connect it whenever they want to log in.
6. ***Efficient to use:*** If the time needed for login is short, then the scheme is considered as efficient to use. Most of the schemes such as OTP, QR code based schemes offer this benefit only partially, while graphical password based schemes do not offer this benefit at all as it takes time to generate and display graphical CAPTCHAs, in addition to the time taken by the user to enter them using a mouse. The proposed scheme is efficient to use as connecting a paired Bluetooth device on an average takes less time (connecting a paired BT device takes an average of nearly 1.5 seconds) than waiting for an OTP or correctly scanning a QR Code. Also, graphical password based and QR Code / Barcode based schemes require user expertise and understanding for entering/scanning the authentication token.
7. ***Infrequent error:*** Most schemes do not offer the infrequent error benefit. A user may not always receive a valid OTP via SMS in time. Complex QR code-based techniques [21] [19] can have frequent errors due to the involvement of multiple QR code scans or matching of geolocations based on IP addresses. In the case of the proposed scheme, other than BT pairing/connecting problems which are very infrequent in the current scenario, there is no other possibility of errors during the authentication.
8. ***Easy recovery:*** A scheme has the easy recovery property if even after the loss of the trusted device, token or secret information, a user account can be recovered easily. Most of the currently utilized schemes inherently provide this benefit but schemes which use a smartphone or hardware token may require some extra efforts for user verification and recovery.
9. ***Accessible:*** Schemes that use visual/graphical password, floating CAPTCHAs, or QR codes cannot be considered accessible to the users (to users with some disability or other physical conditions) who can use the scheme requiring only username and password for login. QR code scanning and graphical password input based schemes cannot be considered accessible as they require visual interactions. Persons with physical impairments such as visual impairment cannot use graphical input based login systems, whereas they can use touch input based login systems, making these schemes not accessible to them. Also, if the use of a scheme depends on the user having a certain level of technical knowledge in using a smartphone, the scheme cannot be considered completely accessible for an average user.
10. ***Negligible cost per user:*** If a scheme needs an OTP or a separate hardware security token it cannot be considered to have a negligible cost per user. However, if it needs an Internet connection on a smartphone (to log in over PC), it can be considered as somewhat having a negligible cost per user because the Internet is generally available on Smartphones.
11. ***Server Compatible:*** Most of the schemes are somewhat server compatible until they need a separate mechanism to make the authentication scheme work such as QR code generation, certificate generation or scanning for credentials or communication through an Android App. In cases when the server needs to implement specific visual appearances (floating CAPTCHA's, dynamic security skins etc.) or when their implementation depends on a specific platform, the scheme is not considered completely server compatible. Google 2 step and our proposed scheme require minimal changes at the server side, which include the generation of OTP and its verification, and fetching of BT properties of the Smartphone from the client end as a username respectively. Password Manager and traditional U-PWD based schemes are completely server compatible.
12. ***Browser compatible:*** Schemes are browser compatible if users do not have to change their browsers or if they do not need

specific modules or software to be installed for the scheme to work. Until apps and extensions supplied by the scheme as a part of the authentication process do not need specific support by the client browsers such as scripting language specific versions or HTML specific versions, they can be considered partially providing this benefit. Schemes that require a specific Flash-based plugin or require the user to install specific modules to his browser to display their content cannot be considered to provide this benefit.

13. ***Mature:*** Only the schemes which are in public use and have been tested aggressively can be considered mature. Schemes which do not have significant variations from the existing authentication schemes can be considered partially mature. E.g. Mukhopadhyay et al. [71] and Dodson et al. [72] proposals are based on QR code scan similar to WhatsApp web application. Our proposed scheme also has small and easily verifiable changes (Bluetooth address fetching and use of AID during authentication) in design from the existing multifactor authentication schemes.
14. ***Non-proprietary***: Schemes which can be used without requiring any explicit approvals from the creators are categorized as non-proprietary.
15. ***Resilient to physical observation:*** A scheme is considered resilient to physical observation if all the authentication credentials cannot be obtained by physical observation (such as shoulder surfing, filming the keyboard, or thermal imaging of the keyboard). Schemes such as Google 2 step, graphical password based schemes etc. do not offer this benefit completely as physical observation techniques can be used to get a complete set of user credentials from the user. The proposed scheme, QR code based and push login based schemes are safe from these attacks as all credentials cannot be acquired by physical observation.
16. ***Target impersonation*** via attackers is also not possible with most of the schemes that use secret encryption keys, QR code scanning via Smartphones, OTP, or hardware tokens for user authentication as an attacker having the knowledge of user personal details (date of birth, age, etc.) cannot get access to the user account (via recovery options or backdoors). The attacker also needs to verify user information via secret keys, hardware security token or via OTP/PIN which is not possible for him/her.
17. ***Throttled and unthrottled guessing***: Password guessing or breaking via throttled or unthrottled guessing is not possible in any schemes except U-PWD and password manager due to the involvement of QR code, hardware tokens, or graphical information, which is difficult to crack by guessing. An easy PWD set for U-PWD and password manager based schemes can be easily guessed by throttled/unthrottled guessing.
18. ***Internal observation:*** Internal observation (via host keylogging or sniffing of communication between client and server via malicious browser extension) is still a security threat for most of the schemes.
19. ***Resilient to leaks from other verifiers***: Schemes which involve a third party during verification do not offer the resilient to leaks from other verifiers benefit.
20. ***Phishing:*** Schemes which are not resilient to traditional or RT/CR MITM are considered vulnerable to phishing (see Table 4.3).
21. ***Resilient to theft***: Almost all schemes are resilient to theft of the hardware token or smartphone except the Dodson et al. and password manager based schemes. In the case of Dodson et al. or password manager based authentication schemes, an attacker with access to the smartphone/PC (respectively) can directly access the user account without entering any user credentials or PIN.
22. ***No trusted third party***: Schemes that utilize a trusted third party during the authentication process does not provide this benefit. Tricipher [68], SAASPASS [70], Mukhopadhyay et al. [71] and password manager use a trusted third party for authentication.
23. ***Explicit consent:*** Almost all authentication schemes need explicit consent for login by the user, except for password managers. Password manager's autofill user credentials when the registered webpage is opened on the browser.
24. ***Unlinkable***: Most of the schemes are unlinkable with the authenticator and cannot be used to link it with the users. However, as Kim et al.'s method [19] uses the IP address, it is somewhat linkable. The IP address can be used to identify that the same user is trying to log in to different websites. The proposed scheme does not offer this benefit completely when the user uses the same BT device to log in to different websites. This is similar to the problem caused by using the same email id as a username while creating accounts with different websites, which makes user accounts easily linkable. Our proposed scheme seems to perform better than the existing schemes based on the comparison shown in Table 4.6.

5. Discussions and conclusions

This paper presents a novel authentication scheme that can handle latest phishing attacks such as RT, CR MITM phishing, MEP attack, and App spoofing. The advantages of the proposed scheme include:

1. Reduction in the number of credentials to be entered and hence remembered by the user. The user only needs to remember the password while other credentials are extracted automatically by the authentication scheme with the user's explicit consent. This provides security against RT MITM attack as an adversary cannot obtain all authentication credentials and relay them in real time.
2. As a Chrome App / Android App is used to enter the password, it is not possible for a malicious browser extension or a malicious peer App to sniff form data or log user password due to the same origin policy, avoiding the possibility of an MEP attack.
3. The use of AID avoids the risk of CR MITM phishing and App spoofing-based attacks.

Also, an advantage of the scheme is that it is client side independent as compared to schemes such as Tricipher. The scheme has been implemented and tested for its efficiency in terms of CPU and memory utilization. Results show a favorable performance. A comparison of it with existing schemes in terms of usability and deployability shows that it performs better than other schemes that provide the same level of security. Also, a single Chrome App can be used to login to multiple websites in the future. The Chrome App will be provided to the user by a trusted third party (any trusted organization or the organization itself). All authentic websites will have to individually store the Bluetooth address of the smartphones of their registered users during the initial registration. Such a modified design of the proposal has been implemented at our lab and is currently going through testing and performance evaluation. We will discuss and present this modification in our future publications.

However, some of the current limitations of the proposed schemes are that the:

- The scheme requires that the user must have a smartphone with him when he is trying to log in on a website using his PC. The survey in [94] shows that worldwide the number of smartphone users till 2017 was around 2.32 billion. It has also

Table 4.5
Comparison with existing schemes using Bonneau et al.'s framework.

		Google 2 Step	SAASPASS	Xie et al.	Kim et al.	Mukhopadhyay et al.	Dodson et al.	Leung et al.	Zhu et al.	Tricipher et al.	RSA SecurID token	Yubikey U2F	Push Login	Password Managers	U-PWD	Proposed Scheme
Usability	1. Memory wise effortless	✗	✗	✗	❍	✗	✓	✗	✗	✗	✗	✗	❍	❍	✗	❍
	2. Scalability for users	✗	✗	✗	✗	✗	✓	❍	❍	❍	❍	❍	✓	✓	✓	✓
	3. Nothing to carry	❍	❍	❍	❍	❍	❍	✓	✓	✗	✗	✗	❍	✓	✓	❍
	4. Physically effortless	❍	❍	✗	✗	✗	✗	✗	✗	✓	❍	❍	❍	✓	✓	✓
	5. Easy to learn	✓	❍	❍	✗	❍	❍	✗	✗	✓	✓	✓	✓	✓	✓	✓
	6. Efficient to use	❍	✓	❍	❍	❍	❍	✗	✗	✓	✓	✓	✓	✓	✓	✓
	7. Infrequent errors	❍	✓	✗	✗	❍	❍	✗	❍	✓	✓	✓	✓	✓	✓	✓
	8. Easy recovery from loss	❍	❍	❍	❍	❍	❍	✓	✓	❍	❍	❍	❍	❍	✓	❍
Deployability	9. Accessible	❍	❍	✗	✗	✗	✗	✗	✗	✓	❍	✓	❍	✓	✓	❍
	10. Negligible cost/user	✗	❍	❍	❍	❍	❍	✗	✓	✓	✗	✗	❍	✓	✓	✓
	11. Server compatible	❍	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	❍
	12. Browser compatible	✓	✓	❍	❍	❍	❍	❍	❍	✓	✓	❍	✓	✓	✓	❍
	13. Mature	✓	✓	✗	✗	❍	❍	✗	✗	❍	✓	✓	✓	✓	✓	❍
	14. Non-Proprietary	✓	❍	✓	✓	✓	✓	✓	✓	✗	✗	✗	❍	✓	✓	✓
Security	15. Resilient to physical observation	❍	❍	✓	✓	✓	✓	❍	❍	❍	❍	✓	❍	❍	✗	❍
	16. Resilient to target impersonation	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓
	17. Resilient to throttled guessing	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓
	18. Resilient to unthrottled guessing	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓
	19. Resilient to internal observation	✗	✗	✗	❍	✗	✗	❍	❍	✗	✗	❍	❍	✗	✗	❍
	20. Resilient to leak from other verifiers	✓	❍	✓	✓	❍	✓	✓	✓	❍	❍	❍	✓	❍	✓	✓
	21. Resilient to Phishing	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗	✓	✗	✓	✗	✓
	22. Resilient to Theft	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✗	✓	✓
	23. No Trusted Third Party	✓	✗	✓	✓	✗	✓	✓	✓	✗	✗	✗	✓	✗	✓	✓
	24. Requiring explicit consent	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓
	25. Unlinkable	✓	✓	✓	❍	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	❍
	Benefit Offered Count	***12***	***10***	***10***	***9***	***8***	***11***	***11***	***12***	***14***	***11***	***13***	***14***	***14***	***18***	***15***

Note: ✓ → Offer the benefit, ✗ → Benefit is not offered, ❍ → Partially offers the benefit, Benefits offered count → Number of benefits completely offered by a given scheme

been predicted [94] that this figure will increase to 2.87 billion by 2020. In one of the survey, it has been reported that around 80% of the Internet users own a smartphone [95]. Based on these surveys it can be assumed that most of the users who use Internet services will be able to use the proposed scheme if launched by the websites.

- The schemes discussed in the paper have not been analyzed for security against host-based malware at this stage. Sophisticated malware capabilities may vary from accessing user keystrokes, system files, data, and resources, intercepting the RAM data via real-time memory dumping and analysis etc. Malware such as keystroke loggers can compromise most of the authentication schemes including OTP/PIN-based schemes. Malware such as screen loggers can compromise graphical password and QR code based schemes while malware having sophisticated capabilities can access secret credentials stored at the TPM module and can also obtain the authentication credentials conveyed through separate hardware tokens via RAM memory dumping and analysis. Malware having access to system storage and the user's windows password or master password at the same time can compromise user passwords stored with Google Chrome password manager [91] and third-party password managers such as LastPass [51] respectively. Hence, we have currently not considered the host-based malware in this research work.

Acknowledgements

The authors happily acknowledge the constructive suggestions and help from Mr. Shashank Arora, SUNY Albany USA, from Miss Niharika Ahuja, Bachelor of Technology student at NIT, Hamirpur, from Ms. Kimberly Oostman, Ph.D. Student, University of New Mexico, USA.

Appendix

- A document describing the reproduction steps of the attacks launched for carrying out the security analysis. https://drive.google.com/open?id=1Wc3uDzgdLMptlS5cUmgp3G4kXMYj4wUOATdybJrPQ3w
- A video created to demonstrate hacking credentials auto-filled by password managers (LastPass) using MEP attack can be accessed here: https://youtu.be/uNwahzlzpSg
- A video created to demonstrate the MITM phishing attack for hacking username and password of a Gmail account can be accessed here: https://youtu.be/nccDJMlkJtc
- A video breaking SASSPASS authentication using RT MITM phishing can be accessed here: https://youtu.be/y-w0RfCaBrQ
- A video demonstrating hacking user accounts through a relay of QR codes used in WhatsApp web authentication can be accessed here: https://youtu.be/6FThk1Iystw
- A video demonstrating keystroke logging, phishing via malicious browser extensions. Also, demonstrates how malicious extensions cannot record the keystrokes on peer apps due to same origin policy: https://youtu.be/fLfzsAodsBw

References

[1] Lastdrager EE. Achieving a consensual definition of phishing based on a systematic review of the literature. Crime Sci 2014;3:9.
[2] Wikipedia *Phishing*. Available: https://en.wikipedia.org/wiki/Phishing#cite_note-50.
[3] Badra M, El-Sawda S, Hajjeh I. Phishing attacks and solutions. In: Proceedings of the 3rd international conference on Mobile multimedia communications; 2007. p. 42.
[4] Dodge RC Jr, Carver C, Ferguson AJ. Phishing for user security awareness. Comput Secur 2007;26:73–80.
[5] Phishing.org. *Phishing Techniques*. Available: http://www.phishing.org/phishing-techniques.
[6] M.T. Banday and J.A. Qadri, "Phishing-A growing threat to e-commerce" *arXiv preprint* arXiv:1112.5732, 2011.
[7] Hong J. The state of phishing attacks. Commun ACM 2012;55:74–81.
[8] PCWorld. *Types of phishing attacks*. Available: http://www.pcworld.com/article/135293/article.html.
[9] Varshney G, Misra M, Atrey PK. A survey and classification of web phishing detection schemes. Secur Commun Netw 2016;9:6266–84 26 OCT 2016.
[10] Varshney G, Misra M, Atrey PK. A phish detector using lightweight search features. Comput Secur 2016;62:213–28.
[11] Islam R, Abawajy J. A multi-tier phishing detection and filtering approach. J Network Comput Appl 2013;36:324–35.
[12] Almomani A, Gupta BB, Atawneh S, Meulenberg A, Almomani E. A survey of phishing email filtering techniques. IEEE Commun Surveys Tut 2013;15:2070–90.
[13] A. Almomani, B.B. Gupta, T.-C. Wan, A. Altaher, and S. Manickam, "Phishing dynamic evolving neural fuzzy framework for online detection zero-day phishing email," arXiv preprint arXiv:1302.0629, 2013.
[14] Gupta BB, Arachchilage NAG, Psannis KE. "Defending against phishing attacks: taxonomy of methods, current issues and future directions,". Telecommun Syst. 2018;67(2):247–67.
[15] Gupta BB, Tewari A, Jain AK, Agrawal DP. "Fighting against phishing attacks: state of the art and future challenges". Neural Comput Appl. 2017;28(12):3629–54.
[16] Ramesh G, Gupta J, Gamya PG. "Identification of phishing webpages and its target domains by analyzing the feign relationship". J Inf Security Appl. 2017;35:75–84.
[17] Gupta S, Pilli ES, Mishra P, Pundir S, Joshi RC. Forensic analysis of E-mail address spoofing. presented at the Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference - Noida, India; 2014.
[18] Latze C. "Stronger Authentication in E-Commerce-How to protect even naive Users against Phishing, Pharming, and MITM attacks," *RVS Retreat 2007 at Quarten*, pp. 111–116, 2007.
[19] Kim S-H, Choi D, Jin S-H, Lee S-H. Geo-location based QR-Code authentication scheme to defeat active real-time phishing attack. In: Proceedings of the 2013 ACM workshop on Digital identity management; 2013. p. 51–62.
[20] Leung C-M. Depress phishing by CAPTCHA with OTP. In: Anti-counterfeiting, Security, and Identification in Communication, 2009. ASID 2009. 3rd International Conference on; 2009. p. 187–92.
[21] Xie M, Li Y, Yoshigoe K, Seker R, Bian J. CamAuth: Securing Web Authentication with Camera. In: High Assurance Systems Engineering (HASE), 2015 IEEE 16th International Symposium on; 2015. p. 232–9.
[22] Aggarwal A, Kumar S, Shah A, Viswanath B, Zhang L, and Kumaraguru P. "Spying Browser Extensions: Analysis and Detection," arXiv preprint arXiv:1612.00766, 2016.
[23] Bandhakavi S, King ST, Madhusudan P, Winslett M. VEX: Vetting Browser Extensions for Security Vulnerabilities. In: USENIX Security Symposium; 2010. p. 339–54.
[24] Guha A, Fredrikson M, Livshits B, Swamy N. Verified security for browser extensions. In: Security and Privacy (SP), 2011 IEEE Symposium on; 2011. p. 115–30.
[25] Dhawan M, Ganapathy V. Analyzing information flow in JavaScript-based browser extensions. In: Computer Security Applications Conference, 2009. ACSAC'09. Annual; 2009. p. 382–91.
[26] Ter Louw M, Lim JS, Venkatakrishnan VN. Enhancing web browser security against malware extensions. J Comput Virol 2008;4:179–95.
[27] Ter Louw M, Lim JS, Venkatakrishnan VN. Extensible web browser security. In: International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment; 2007. p. 1–19.
[28] Utakrit N. Review of browser extensions, a man-in-the-browser phishing techniques targeting bank customers. In: 7th Australian Information Security Management Conference; 2009. p. 110–19.
[29] APWG. *APWG Phishing Attack Trends Reports*. Available: http://www.antiphishing.org/resources/apwg-reports/.
[30] APWG. *APWG phishing attack trends reports*. Available: http://www.antiphishing.org/resources/apwg-reports/.
[31] APWG. *APWG phishing attack trends reports*. Available: http://www.antiphishing.org/resources/apwg-reports/.
[32] DELLEMC. *Rsa online fraud resource center*. Available: http://www.emc.com/onlinefraud.
[33] OWASP. *Qrljacking*. Available: https://www.owasp.org/index.php/Qrljacking.
[34] Lu Y, Li L, Peng H, Yang Y. An energy efficient mutual authentication and key agreement scheme preserving anonymity for wireless sensor networks. Sensors 2016;16:837.
[35] Lu Y, Li L, Peng H, Yang Y. A secure and efficient mutual authentication scheme for session initiation protocol. Peer-to-Peer Networking Appl March 01 2015,;9:449–59.
[36] Kapravelos A, Grier C, Chachra N, Kruegel C, Vigna G, Paxson V. Hulk: Eliciting Malicious Behavior in Browser Extensions. In: USENIX Security; 2014. p. 641–54.
[37] Saini A, Gaur MS, Laxmi V, Conti M. Colluding browser extension attack on user privacy and its implication for web browsers. Comput Secur 2016;63:14–28.
[38] Saini A, Gaur MS, Laxmi V, Singhal T, Conti M. Privacy Leakage Attacks in Browsers by Colluding Extensions. presented at the Information Systems Security: 10th International Conference, ICISS 2014 Hyderabad, India; 2014.

[39] Shahriar H, Weldemariam K, Zulkernine M, Lutellier T. Effective detection of vulnerable and malicious browser extensions. Comput Secur 2014;47:66–84.
[40] Varshney G, Misra M, Atrey P. Detecting Spying and Fraud Browser Extensions: Short Paper. In: Proceedings of the 2017 on Multimedia Privacy and Security; 2017. p. 45–52.
[41] Center MMP *Browser extension hijacks Facebook profiles*. Available: https://blogs.technet.microsoft.com/mmpc/2013/05/10/browser-extension-hijacks-facebook-profiles/.
[42] Varshney G, Misra M, Atrey P. Browshing - a new way of phishing using a malicious browser extension. presented at the International Conference on Innovations in Power and Advanced Computing Technologies (IPACT 17) (In Press) Vellore, Chennai, India; 2017.
[43] Barth A, Felt AP, Saxena P, Boodman A. Protecting Browsers from Extension Vulnerabilities. In: Proceedings of the Network and Distributed System Security Symposium, NDSS 2010; 2009. p. 1–12.
[44] Zhu BB, Yan J, Bao G, Yang M, Xu N. Captcha as Graphical Passwords-A New Security Primitive Based on Hard AI Problems. IEEE Trans Inf Forensics Secur 2014;9:891–904.
[45] Fraser N *The usability of picture passwords*. Available: http://www.tricerion.com/wp-content/uploads/2013/09/Usability-of-picture-passwords.pdf.
[46] Dhamija R, Tygar JD. The battle against phishing: Dynamic Security Skins. In: Proceedings of the 2005 symposium on Usable privacy and security; 2005. p. 77–88.
[47] Lu Y, Li L, Peng H, Yang Y. An anonymous two-factor authenticated key agreement scheme for session initiation protocol using elliptic curve cryptography. Multimedia Tools Appl. January 01 2015;76:1801–15.
[48] Barker I *Saaspass makes two-factor authentication available to the masses*. Available: https://betanews.com/2015/01/15/saaspass-makes-two-factor-authentication-available-to-the-masses/,.
[49] Google. *Stronger security for your google account*. Available: https://www.google.com/landing/2step.
[50] NetIQ. *Hard Tokens vs. Soft Tokens: Why Soft Tokens Are the Better Option*. Available: https://www.netiq.com/docrep/documents/xvbozdzzxj/hard_tokens_vs_soft_tokens_fpp.pdfn.
[51] Lastpass. *Lastpass*. Available: https://www.lastpass.com/.
[52] Urbanairship. *Push Notifications Explained*. Available: https://www.urbanairship.com/push-notifications-explained.
[53] Varshney G, Misra M, Atrey P. Push Notification Based Login Using BLE Devices. 2nd International Conference on Information Technology, Information Systems, and Electrical Engineering Yogyakarta,Indonesia; 2017.
[54] Google. *Sign in faster with 2-Step Verification phone prompts*. Available: https://support.google.com/accounts/answer/7026266?co=GENIE.Platform%3DiOS&hl=en.
[55] Suo X, Zhu Y, Owen GS. Graphical passwords: A survey. In: Computer security applications conference, 21st annual; 2005. p. 10–472.
[56] Suresh S, Gopalakrishnan P. "On reviewing the limitations of graphical password scheme". Journal of Computer Science and Engineering Research 2014;1:31–5.
[57] Dhillon PK, Kalra S. A lightweight biometrics based remote user authentication scheme for IoT services. J Inf Security Appl 2017;34:255–70.
[58] Mahfouz A, Mahmoud TM, Eldin AS. A survey on behavioral biometric authentication on smartphones. J Inf Secur Appl 2017;37:28–37.
[59] Jacob AJ, Bhuvan NT, Thampi SM. Feature level fusion using multiple fingerprints. Comput Sci-New Dimens Perspect 2011;4:13–18.
[60] Erden M *Advantages and Disadvantages of Biometrics Authentication*. Available: http://www.sestek.com/2016/11/advantages-disadvantages-biometric-authentication/.
[61] Kokalitcheva K *Fingerprint Spoofing Is Much Easier Than You Think*. Available: http://fortune.com/2016/02/24/fingerprint-spoofing-easy/.
[62] Newman LH *Hackers Trick Facial-Recognition Logins With Photos From Facebook (What Else?)*. Available: https://www.wired.com/2016/08/hackers-trick-facial-recognition-logins-photos-facebook-thanks-zuck/.
[63] Yubico. *Yubico FIDO U2F*. Available: https://www.yubico.com/solutions/fido-u2f/.
[64] Beal V *RSA Secure ID*. Available: http://www.webopedia.com/TERM/R/rsa_secure_id.html.
[65] DUO. *Security Tokens*. Available: https://duo.com/product/trusted-users/two-factor-authentication/authentication-methods/security-tokens.
[66] Cox J *Hackers Show Proofs of Concept to Beat Hardware-Based 2FA*. Available: https://motherboard.vice.com/en_us/article/8xazek/hackers-show-proof-of-concepts-to-beat-hardware-based-2fa.
[67] Team Y *Infineon RSA Key Generation Issue*. Available: https://www.yubico.com/2017/10/infineon-rsa-key-generation-issue/.
[68] TRICIPHER, "Preventing man in the middle phishing attacks with multi-factor authentication," 2016.
[69] Google. *Using security key for 2-step verification*. Available: https://support.google.com/accounts/answer/6103523.
[70] SAASPASS. *Two-factor authentication with proximity uses ibeacon bluetooth low energy (ble) to authenticate users instantly*. Available: https://saaspass.com/technologies/proximity-instant-login-two-factor-authentication-beacon.html.
[71] Mukhopadhyay S, Argles D. An Anti-Phishing mechanism for single sign-on based on QR-code. In: Information Society (i-Society), 2011 International Conference on; 2011. p. 505–8.
[72] Dodson B, Sengupta D, Boneh D, Lam MS. Secure, consumer-friendly web authentication and payments with a phone. In: *International Conference on Mobile Computing, Applications, and Services*, Santa Clara; 2010. p. 17–38.
[73] Lu Y, Li L, Peng H, Yang Y. A Novel smart card based user authentication and key agreement scheme for heterogeneous wireless sensor networks. Wireless Personal Commun. September 01 , 2017;96:813–32.
[74] Ross B, Jackson C, Miyake N, Boneh D, Mitchell JC. Stronger Password Authentication Using Browser Extensions. In: Usenix security; 2005. p. 17–32.
[75] Rouse M *two-factor authentication (2FA)*. Available: http://searchsecurity.techtarget.com/definition/two-factor-authentication.
[76] Eliasson C, Fiedler M, rstad IJÃ. Evaluation of authentication schemes based on security, user-friendliness and complexity. EuroNF IA. 7.5 Workshop in Socio-Economic Issues of Networks of the Future; 2008.
[77] Xu Z, Zhu S. Abusing notification services on smartphones for phishing and spamming. In: 6th USENIX conference on Offensive Technologies (WOOT'12); 2012. p. 1–11.
[78] Bright P *RSA finally comes clean: SecurID is compromised*. Available: https://arstechnica.com/information-technology/2011/06/rsa-finally-comes-clean-securid-is-compromised/.
[79] Yubico. *Fido u2f (universal 2nd factor)*. Available: https://www.yubico.com/about/background/fido/.
[80] Das A, Ghose A, Razdan A, Saran H, Shorey R. Enhancing performance of asynchronous data traffic over the Bluetooth wireless ad-hoc network. In: INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE; 2001. p. 591–600.
[81] Chrome G *chrome.bluetooth*. Available: https://developer.chrome.com/apps/bluetooth.
[82] Malisa L, Kostiainen K, Capkun S. Detecting mobile application spoofing attacks by leveraging user visual similarity perception. In: Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy; 2017. p. 289–300.
[83] Wikipedia. *Salt (cryptography)*. Available: https://en.wikipedia.org/wiki/Salt_(cryptography).
[84] Wikipedia. *Cryptographically secure pseudorandom number generator*. Available: https://en.wikipedia.org/wiki/Cryptographically_secure_pseudorandom_number_generator#cite_note-16.
[85] Redondo A *AnotherMonitor*. Available: https://play.google.com/store/apps/details?id=org.anothermonitor&hl=en.
[86] Werner S, Hauck C, Masingale M. Password Entry Times for Recognition-based Graphical Passwords. In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting; 2016. p. 755–9.
[87] Wikipedia. *Man-in-the-browser*. Available: https://en.wikipedia.org/wiki/Man-in-the-browser.
[88] RSA, "Making Sense of Man In the Browser Attacks," RSA Security LLC2011.
[89] Kasiviswanathanblog *Change Bluetooth Address*. Available: https://kasiviswanathanblog.wordpress.com/2017/03/28/change-bluetooth-address/.
[90] Yahoo. *Yahoo Sign In*. Available: https://login.yahoo.com/.
[91] Google. *Manage saved passwords*. Available: https://support.google.com/chrome/answer/95606?co=GENIE.Platform%3DDesktop&hl=en.
[92] Bonneau J, Herley C, Van Oorschot PC, Stajano F. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In: Security and Privacy (SP), 2012 IEEE Symposium on; 2012. p. 553–67.
[93] Bonneau J, Preibusch SR. The Password Thicket: Technical and Market Failures in Human Authentication on the Web. In: WEIS. USA: Harvard University; 2010. p. 1–48.
[94] Statista. *Number of smartphone users worldwide from 2014 to 2020 (in billions)*. Available: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/.
[95] Sukhraj R *31 Mobile Marketing Statistics to Help You Plan for 2017*. Available: https://www.impactbnd.com/blog/mobile-marketing-statistics-for-2016.