Usability and Security Analysis of the Compare-and-Confirm Method in Mobile Push-Based Two-Factor Authentication

Mohammed Jubur, Nitesh Saxena, *Senior Member, IEEE*, and Faheem A. Reegu

***Abstract*—Push-based two-factor authentication (2FA) methods, such as the "Just-Confirm" approach, are popular due to their user-friendly design, requiring users to simply approve or deny a push notification on their mobile device. However, these methods are vulnerable to "concurrency attacks," where an attacker attempts to log in immediately after the legitimate user, causing multiple push notifications that may lead to users inadvertently approving fraudulent access. This vulnerability arises because the login notifications are not uniquely bound to individual login attempts. To address this issue, Push-Compare-and-Confirm 2FA method enhances security by associating each login notification with a unique code displayed on both the authentication terminal and the push notification. Users are required to match these codes before confirming access, thereby binding the notification to a specific login attempt. Recognizing the ubiquity of mobile devices in daily life, we conducted a comprehensive user study with 65 participants to evaluate the usability and security of Push-Compare-and-Confirm. The study considered two scenarios: one where the user's second-factor device (phone) is physically separate from the authentication terminal (e.g., logging in on a PC and confirming on the phone), and another where the phone serves as both the authentication terminal and the second-factor device. Participants completed 24 login trials, including both benign and attack scenarios, with varying code lengths (four characters and six characters). Our results indicate that while Push-Compare-and-Confirm maintains high usability in benign scenarios, with True Positive Rates (*TPR*) exceeding 95%, it presents significant challenges in attack detection. Participants correctly identified only about 50% of fraudulent login attempts, indicating a substantial vulnerability remains. These findings suggest that although Push-Compare-and-Confirm enhances security over standard push-based 2FA methods, additional measures—such as more intuitive interface designs, clearer visual cues, and user education on the importance of code verification—are necessary to improve attack detection rates without compromising usability.**

***Index Terms*—Two-factor authentication (2FA), push-based authentication, push-compare-and-confirm (PushCC), security, usability, concurrency attacks, mobile security, authentication protocols, usability evaluation, code comparison, authentication security.**

I. Introduction

PUSH-BASED two-factor authentication (Push-2FA), especially prominent in mobile-centric security scenarios, is championed by platforms such as Duo [1], Google [2], LastPass [3], and Microsoft [4] to improve the usability of two-factor authentication (2FA) by simplifying the process. In Push-2FA, users only need to tap once on their smartphone to complete the authentication process, unlike traditional 2FA, which requires entering a one-time PIN (OTP).

Traditional 2FA verifies the possession of the second-factor device, such as a smartphone, using an OTP sent via SMS or generated by an app on the user's phone. The OTP is then manually entered into the authentication terminal. In contrast, Push-2FA verifies the possession of the second-factor device by sending a push notification, or login notification, to the device running a pre-installed 2FA app. Users are prompted to approve or deny the authentication attempt by tapping a button on their phone.

A notable feature of Push-2FA is its ability to include the IP geolocation of the login attempt within the notification. This is evident in systems like Duo and Google Push-2FA (Fig. 2). The user should carefully evaluate this information before approving or denying the login attempt. However, LastPass Push-2FA does not provide this IP geolocation information, and there is no indication on the service site for the user to evaluate the login notification. This absence of information prevents users from verifying the legitimacy of the login attempt based on location data, thereby increasing the risk of a vulnerability known as a *concurrency attack* as identified by Jubur et al. [5]. Without the ability to evaluate the IP address or location, users may inadvertently approve fraudulent login attempts.

In contrast, Microsoft Push-2FA employs a different design referred to as "Push-Compare-and-Confirm." In this design, a unique code is generated and displayed simultaneously on the authentication terminal and within the login notification. The user is prompted to compare the codes on both devices and approve or deny the authentication attempt only if they match, referred to as *Push-Compare-and-Confirm*. Fig. 1 illustrates how Push-Compare-and-Confirm works. Though this method enhances security, it introduces an added step for users, possibly

Received 23 October 2023; revised 12 November 2024; accepted 18 December 2024. Date of publication 30 December 2024; date of current version 7 May 2025. This work was supported by the University of Alabama at Birmingham (UAB) Institutional Review Board under Application IRB-300000222. Recommended for acceptance by A. A. Nayak.

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by (Name of Review Board or Committee) (IF PROVIDED under Application No. xx, and performed in line with the (Name of Specific Declaration)).

The authors are with the Department of Computer, Network Engineering, Jazan University, Jazan 45142, Saudi Arabia, and also with the Department of Computer Science, Engineering, Texas A&M University, College Station, TX 77840 USA (e-mail: mjabour@jazanu.edu.sa; nsaxena@tamu.edu; freegu@jazanu.edu.sa).

This article has supplementary downloadable material available at https://doi.org/10.1109/TMC.2024.3524093, provided by the authors.

Digital Object Identifier 10.1109/TMC.2024.3524093

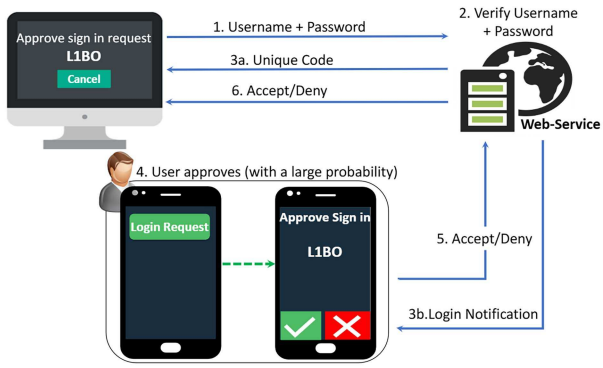


Fig. 1. During normal operation of Push-Compare-and-Confirm, the web service sends a login notification containing a unique code to the user's phone. Simultaneously, the same unique code is displayed on the authentication terminal (e.g., desktop PC). The user compares the codes on both devices and approves the login notification if the codes match, thereby successfully authenticating and accessing the web service.

affecting usability. This research seeks to critically evaluate the efficacy and user experience of the Push-Compare-and-Confirm method in 2FA scenarios.

By 2029, 98% of Americans will have access to the internet, up from 94.6% today [6]. The widespread connectivity highlights the crucial role of internet-based technologies in daily life. Concurrently, mobile devices have become the dominant means of accessing the internet. In Q1 2023, mobile devices (excluding tablets) accounted for 58.33% of global website traffic, a substantial increase from 31.16% in Q1 2015, reflecting a 75% rise since 2015. Additionally, 95.8% of individuals use mobile phones to access the internet, compared to 62.9% using laptops or desktops. Mobile internet activity now drives up to 75% of E-Commerce visits, and mobile apps have surpassed desktops in web page views by 5.64% as of 2021 [7]. These trends highlight a significant shift towards mobile devices not only as primary access points to the internet but also as essential tools for managing online accounts.

Given the widespread use of mobile devices and their continually advancing capabilities, the most interesting aspect of the smartphone is its ability to serve as both an authentication terminal and a second-factor device. This paper studies the usability and security of Push-Compare-and-Confirm in two settings. The first setting, referred to as the PC setting, involves the user deploying a second-factor device physically separated from the authentication terminal. The second setting is when the user utilizes the second-factor device as the authentication terminal.

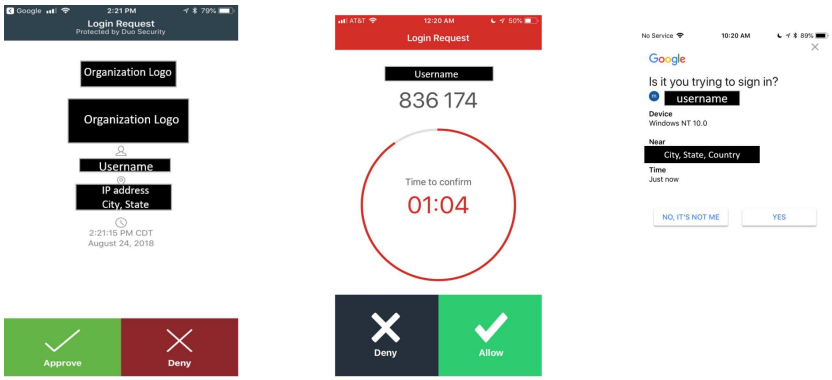
The hypothesis is that Push-Compare-and-Confirm offers better security compared to the standard ''just confirm'' Push-2FA. In the phone setting, it is expected that users will make more errors and take more time to complete the authentication process due to the need to keep switching between the authentication app and the browser app to verify the codes. In contrast, the PC setting allows users to easily compare the unique code displayed on the authentication terminal (PC) with the code on their second-factor device (smartphone), as both devices can be viewed simultaneously without app-switching or screen toggling.

Our Contributions: In this paper, we present a comprehensive study on the usability and security of the Push-Compare-and-Confirm two-factor authentication method. We designed and conducted a user study with **65 participants** to evaluate Push-Compare-and-Confirm in terms of True Positive Rate (*TPR*) in benign settings and False Positive Rate (*FPR*) in adversarial settings. Participants were instructed to simulate login processes to an online account using our implementation of the Push-Compare-and-Confirm system. In the process, they responded to login notifications generated on their second-factor devices (phones) to prove possession.

Our Push-Compare-and-Confirm design closely mimics Microsoft's existing 2FA framework by presenting login codes with bold fonts, ensuring consistency and familiarity for users. We introduced two unique code lengths—four characters (Fig. 3(a)) and six characters (Fig. 3(b))—to assess their impact on usability and security. The study considered two scenarios: one where the phone serves as the second-factor device separate from the authentication terminal (PC setting), and another where the phone functions as both the authentication terminal and the second-factor device (phone setting). This work makes several key contributions:

- We provide the first comprehensive user study evaluating the Push-Compare-and-Confirm method's effectiveness in both benign and adversarial settings, involving a substantial participant pool of 65 individuals.
- We explore the impact of device type (PC vs. phone) and code length (four characters vs. six characters) on the usability and security of Push-Compare-and-Confirm, offering nuanced insights into how these factors influence user performance and system security.
- Our study highlights critical vulnerabilities in the Push-Compare-and-Confirm method, particularly in attack detection, and underscores the need for improved interface designs and user education to enhance security without compromising usability.
- We contribute to the body of knowledge by comparing Push-Compare-and-Confirm with standard push-based 2FA methods, demonstrating that while Push-Compare-and-Confirm improves attack detection rates, significant challenges remain.
- We provide actionable recommendations for enhancing the Push-Compare-and-Confirm method, including interface optimizations and adaptive security measures, informing the future development of more secure and user-friendly authentication systems.

Summary of Key Results: Our results show that the Push-Compare-and-Confirm authentication system performs well in benign sessions, with high *TPR* of 98.48% in the PC setting and 95.76% in the phone setting. However, the *FPR* in attack sessions was approximately 49.5% across both settings, indicating significant challenges in attack detection. While Push-Compare-and-Confirm shows an improvement over standard push-based 2FA methods—in which prior research reported lower attack detection rates—the high *FPR* suggests that the

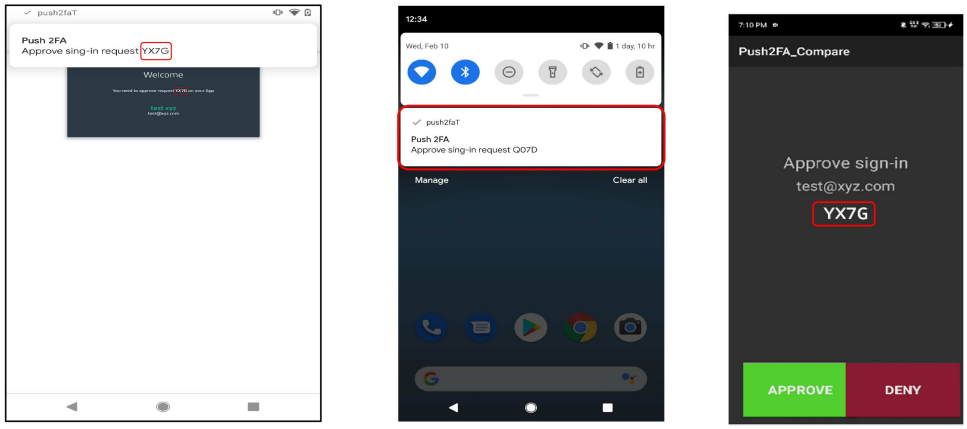


(a) Duo Push phone UI (b) LastPass phone UI (c) Google phone UI

Fig. 2. Push-2FA.

(a) Browser Interface Displaying a four characters Code.

(b) Browser Interface Displaying a six character.



(c) Popup Notification Triggered by Browser During Login Attempt.

(d) Login Notification on Second-Factor Device.

(e) Authentication App Interface.

Fig. 3. User interfaces and notification designs implemented and evaluated in the Push-Compare-and-Confirm two-factor authentication study.

system may not be fully secure against concurrency attacks and fatigue attacks [8]. We found no significant differences in *TPR* or *FPR* between different code lengths or between PC and phone settings.

II. Background and Related Work

A. System Models

Push-2FA is a method of authentication where the user's smartphone receives a push notification that prompts user's response during the authentication process by *approving* or *denying* the login attempt. Push-2FA requires a confidential communication channel between the service and the user device to ensure the push notifications are sent to the correct device. Therefore, the user is required to install the service app on his device to be able to use this method. Some services like Facebook [9], and Google [10] enable Push-2FA in their original apps, so the user will not need to install an additional app to use this method with these services. On the other hand, Duo [1], LastPass [3], and Microsoft [11] required the user

to install the authenticator app of these services to use this method. The authenticator app is a software token that is used to generate the OTP based on the secret shared between the service and the user's device. The login notifications are pushing to the user's device via cloud messaging services like Firebase Cloud Messaging (FCM) [12], Apple Push Notification Service (APNs) [13], and Windows Notification Service (WNS).

In the Push-2FA method, the user first provides his credentials to the service. The service then sends a login notification, also called a push notification, to the user's phone. The purpose of this login notification is to confirm with the user if they initiated the login attempt, or if it was someone else attempting to log in on their behalf. To successfully login to the service, the user needs to approve the push notification by tapping the 'Approve' button. Unlike traditional OTP that is sent via an SMS or generated by the authenticator app, Push-2FA does not require the user to copy the OTP from the phone to the authentication terminal, just tapping a button on the phone is sufficient to login. This makes the Push-2FA scheme more usable compared to the traditional OTP 2FA scheme.

To enhance the security of Push-2FA, the Push-Compare-and-Confirm method introduces a unique code displayed simultaneously on both the authentication terminal and the user's smartphone. Before approving the login notification, the user must verify that the unique codes match. If the codes do not align, indicating a potential fraudulent attempt, the user can deny the login notification and/or report the incident to the service provider. Additionally, Push-2FA incorporates a "time-out" period, within which the user must respond to the push notification. This period typically ranges from 30 seconds to a few minutes, depending on the service provider's policy to ensure timely user response and mitigate delayed attack attempts. If the user fails to respond within this timeframe, the login attempt is automatically denied.

B. Threats in Push-2FA

Push-Compare-and-Confirm, while enhancing security over standard Push-2FA methods, is not impervious to vulnerabilities. Understanding these threats is essential for developing robust authentication systems. The primary vulnerabilities associated with Push-Compare-and-Confirm include concurrency attacks [5], real-time phishing attacks [14], [15], [16], [17], and fatigue attacks [8].

Concurrency Attacks: These attacks occur when an adversary attempts to log in immediately after a legitimate user, resulting in the user receiving multiple push notifications in quick succession. The user may become confused and inadvertently approve the attacker's request, mistaking it for their own legitimate login attempt. The core issue is the absence of distinct identifiers for each login attempt, making it challenging for users to discern which notification corresponds to their action.

Real-time Phishing Attacks: In these attacks, malicious actors deceive users into interacting with fraudulent websites or compel them to divulge their login credentials. For Push-Compare-and-Confirm, the threat lies in users inadvertently responding to fake login notifications, thereby granting attackers unauthorized access. Ensuring that users can reliably distinguish genuine notifications from malicious ones is a significant challenge.

Fatigue Attacks: These attacks exploit users' attentiveness by bombarding them with frequent authentication requests. Over time, users may become complacent and start approving notifications without thoroughly verifying the details, such as the unique codes. This reduction in attention increases the likelihood of users approving fraudulent login attempts.

C. Mitigation Strategies for Push-2FA Vulnerabilities

Addressing the aforementioned vulnerabilities requires strategic enhancements to the Push-Compare-and-Confirm method. Several mitigation strategies have been proposed and implemented in existing systems.

Push-Select-and-Confirm: This strategy specifically mitigates concurrency attacks by requiring users to select the matching unique code on their phone [18]. This additional verification step provides an extra layer of security by ensuring that the user consciously verifies each login attempt. However, Push-Select-and-Confirm remains susceptible to real-time phishing attacks, as attackers may still craft convincing fake notifications that mimic legitimate ones.

Op-2FA: Op-2FA [19] enhances the security of the compare-and-confirm scheme by ensuring the consistency of the checksum (unique code) and integrating the QR code method for verification. A significant advantage of Op-2FA is its resilience against real-time phishing attacks, as it leverages the service's domain name to create a secure login password, making it difficult for attackers to fabricate legitimate-looking notifications. However, Op-2FA may introduce additional complexity for users, potentially impacting usability.

Enhanced User Interfaces and Education: Improving the user interface to provide clearer visual cues and better code comparison tools can help mitigate fatigue attacks by making it easier for users to verify the authenticity of each login attempt. Additionally, educating users about the importance of verifying unique codes and recognizing phishing attempts can significantly reduce the risk of accidental approvals of fraudulent requests.

D. Related Prior Work

Several studies have explored the usability and security aspects of two-factor authentication methods, including Push-2FA and its variations.

Colnago et al. [20] conducted two surveys at Carnegie Mellon University (CMU) following the deployment of the Duo 2FA system. The first survey targeted faculty and staff, while the second focused on students. The surveys assessed users' perceptions of usability and security, revealing that participants believed 2FA added security but found it to be somewhat inconvenient due to the extra steps involved. In contrast, our study employs a Push-2FA design similar to Duo's implementation and measures authentication efficiency by recording the time taken for participants' login attempts. This provides quantitative data on usability, addressing the need identified in prior research.

Reese et al. [21] performed a usability lab study comparing five two-factor authentication methods—SMS, TOTP, Push-2FA, pre-generated codes, and U2F—with 72 participants. They discovered that Push-2FA and U2F had the fastest authentication times, while Push-2FA and TOTP received the highest System Usability Scale (SUS) scores compared to other methods. However, their study did not consider scenarios where users employed their smartphones as both authentication terminals and second-factor devices. Our research addresses this gap by evaluating Push-Compare-and-Confirm under both device configurations and incorporating adversarial settings to assess security robustness.

Uzun et al. [22] analyzed the usability of secure pairing methods, including "compare-and-confirm" and "select-and-confirm". In the former, users compare two-digit sequences displayed on separate devices, while the latter requires users to select the matching number from a group. They found that both methods were easy to use but perceived as less secure. In our study, we enhance the security of the compare-and-confirm approach by using unique codes that combine numbers and characters. This increases the complexity of the codes, reducing the likelihood of successful guessing attacks.

While existing research has significantly advanced the understanding of Push-2FA systems' usability and security, several gaps remain. Notably, there is limited empirical evidence on the effectiveness of enhanced methods like Push-Compare-and-Confirm in real-world scenarios, especially concerning their performance under adversarial conditions and varying device configurations. Additionally, prior studies have not thoroughly explored the impact of unique code lengths and compositions on both usability and security.

Our current study addresses these gaps by conducting a comprehensive user evaluation of the Push-Compare-and-Confirm method across different device settings and code configurations. By simulating both benign and attack scenarios, we assess the method's True Positive Rate (*TPR*) and False Positive Rate (*FPR*), providing a nuanced understanding of its strengths and limitations.

III. Study Design

In this section, we detail the methodology employed to evaluate the Push-Compare-and-Confirm authentication scheme. Our study focuses on assessing the performance, security effectiveness, and user behavior associated with Push-Compare-and-Confirm across various scenarios, including benign and attack sessions, different device types (PC vs. phone), and code lengths (four characters vs. six characters). By analyzing authentication time, accuracy rates, and user consistency and collecting qualitative feedback, we aim to provide comprehensive insights into the usability and security implications of Push-Compare-and-Confirm for real-world application scenarios.

A. Study Objective

This study aims to evaluate the performance, security potential, and user behavior associated with the Push-Compare-and-Confirm authentication method in a lab-based setting. We analyze the time taken for authentication, the True Positive Rate (*TPR*) of successful logins, the False Positive Rate (*FPR*) of failed logins, and user consistency across different authentication scenarios, including both benign and attack sessions. Specifically, we investigate the impact of different factors—such as code length and device type—on Push-Compare-and-Confirm's performance, security effectiveness, and user interaction. By conducting this evaluation, we provide insights into the usability, security implications, and user engagement with Push-Compare-and-Confirm in real-world applications.

B. Implementation

To evaluate Push-Compare-and-Confirm, we implemented an instance of a push-based two-factor authentication system that emulates real-world Push-2FA services. Our implementation consists of two main components:

1) *WebService and Browser Apps:* The WebService-App and Browser-App are integral parts of our Push-2FA system, responsible for user registration, authentication, and login approval processes. The WebService-App, running on a remote web server, utilizes HTML, JavaScript, CSS, and PHP to provide a seamless user experience. It acts as the central hub for the authentication process.
The WebService-App performs the following tasks: upon receiving the user's login credentials from the Browser-App's login form, it verifies their validity. It sends a push notification to the second-factor device (authentication app on the Phone) requesting approval for the login attempt. This push notification, including a unique code embedded within it, establishes a secure communication channel between the user's devices. We utilized Firebase Cloud Messaging (FCM) to facilitate reliable push notification delivery across various platforms. The WebService-App interacts with a MySQL database to securely store and retrieve user account information.
The Browser App, operating on the client machine, serves as the primary interface through which users interact with the Push-2FA system. This application encompasses both sign-up and login forms, facilitating user registration and authentication processes. During the sign-up phase, users create an account by providing the necessary credentials, effectively simulating the registration procedures commonly employed by online services. The Browser App communicates securely with the WebService backend to store and manage user account information, ensuring data integrity and confidentiality.
During the authentication process, the client (browser) collects the user's login credentials and transmits them to the WebService for verification. The unique code embedded in the push notification is presented by the authentication app on the phone. Based on the user's response—either "Approve" or "Deny"—the browser relays this decision back to the web service. If the user approves the login request, the client displays a success message and redirects the user to the intended service (e.g., Gmail login page). Conversely, if the user denies the request, the browser

shows an error message and navigates the user back to the Push-2FA login interface.

2) *Phone-App:* The authentication App (Phone-App), designed for Android devices, functions as a Push-2FA application and serves as the second-factor device. Operating in the background, it generates push notifications to request login confirmations. When a user initiates a login attempt via the browser, the app receives the unique code from the WebService and prompts the user with a push notification to approve or deny the authentication request.

Set-Up Specifics: We utilized the Apache HTTP Server provided by XAMPP [23], an open-source, cross-platform web-server solution, to host the WebService. The study was conducted using a desktop computer as the authentication terminal, Google Chrome as the browser for launching the Browser-App, and a smartphone running Android versions 10-13 as the second-factor device, on which the Phone-App was installed.

Unique Code Design: We implemented two variants of the unique code: **four characters** and **six characters**. In the four characters design, users are presented with a four-character alphanumeric code (Fig. 3(a)). Conversely, the six characters design displays a six-character code (Fig. 3(b)). These variations were employed to assess the impact of code length on usability and security within the Push-Compare-and-Confirm framework.

We implemented two unique code designs: four characters and six characters. The four-character codes consist of both numbers and letters, while the six-character codes are longer to increase complexity. This variation allows us to evaluate the impact of code length on user experience, authentication efficiency, and security. Shorter codes may offer faster authentication, whereas longer codes provide a larger search space, enhancing security against brute-force attacks. Assessing both code lengths helps us understand the trade-off between usability and security in the Push-Compare-and-Confirm method.

While this design choice ensures consistency and familiarity for users, it may also inherit vulnerabilities associated with similar frameworks, as discussed in the **Discussion** section.

C. Login Sessions

To evaluate the performance of participants using our Push-2FA implementation, we incorporated two types of login sessions: benign sessions and attack sessions.

- *Benign Session:* In a benign login session, our Push-2FA system triggers a single push notification containing a unique code. The authentication terminal displays the same unique code that is sent to the user, simulating a real-world login scenario where the user attempts to log in using Push-Compare-and-Confirm. This session type assesses the performance of the Push-Compare-and-Confirm method under normal operating conditions.
- *Attack Session:* In our Push-2FA implementation, attack sessions simulate fraudulent authentication attempts by deliberately creating a code mismatch: the unique code displayed on the authentication terminal differs from the one sent to the user's second-factor device (see Fig. 4). Participants must approve or deny the login based on these mismatched codes, testing Push-Compare-and-Confirm's resistance to unauthorized access.

  This discrepancy assesses users' attentiveness in verifying authentication details and their ability to reject fraudulent requests. It evaluates Push-Compare-and-Confirm's robustness against common attacks like phishing and man-in-the-middle attacks, where attackers trick users into approving unauthorized access. Attack sessions also simulate concurrency and fatigue attacks by sending multiple or repeated authentication requests to confuse or wear down users' vigilance. By randomizing and balancing these attack sessions with benign ones, we prevent participants from predicting attacks, ensuring that responses are based on real-time decisions

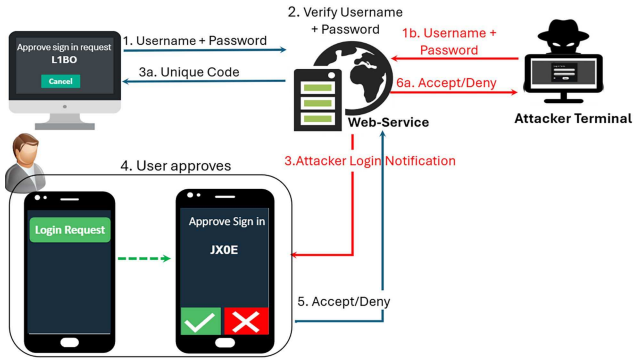


Fig. 4. During the login process, the authentication terminal displays **L1BO** while the push notification on the user's device shows a different code, **JX0E**. The user is expected to notice this discrepancy and deny the login attempt, preventing unauthorized access.

We programmed our Push-2FA system so that each participant received a fixed set of benign and attack login sessions. Specifically, each participant was subjected to a total of 24 login sessions: twelve in the PC setting and twelve in the phone setting. Each setting consists of six benign sessions and six attack sessions. Within these six sessions, three used four characters codes, and three used six characters codes. The order of login sessions was *randomized* to remove any learning biases. Fig. 9 in the Appendix depicts the hierarchical flow of our study's various types of login sessions.

Rationale for Session Types and UI Design: The rationale behind choosing these specific session types is to understand the effectiveness of the Push-Compare-and-Confirm method in both benign and attack scenarios. By comparing performance metrics such as authentication time, True Positive Rate (*TPR*), False Positive Rate (*FPR*), and user consistency between the benign and attack sessions, we can assess the system's ability to differentiate between valid and invalid login attempts.

Our UI design choices, specifically the use of bold fonts for presenting login codes and the variation in code lengths, directly influence key performance metrics (see Fig. 10(a)). The use of bold fonts serves to differentiate the unique code from the surrounding text in the notification message, enhancing the visibility of legitimate codes and potentially increasing *TPR* by making it easier for users to identify and approve legitimate

login attempts correctly. This design is similar to Microsoft's approach in the Push-Compare-and-Confirm method. Conversely, suppose the bold styling does not sufficiently help users differentiate between legitimate and illegitimate codes in adversarial scenarios. In that case, it may lead to an increase in *FPR*, where users inadvertently approve fraudulent login attempts. This relationship between UI design and performance metrics is further explored in the **Discussion** section.

D. Evaluation Methodology

To evaluate the effectiveness of the Push-Compare-and-Confirm authentication scheme from the perspectives of usability and security, we conducted a formal lab-based study. This study aimed to quantify several key metrics, which are detailed below:

- *Authentication Time: How long does it take for participants to complete the authentication process?* The *starting point* is when the system begins verifying the participant's first factor (i.e., username and password), and the *ending point* is when the system receives the response to the login notification. Measuring this duration helps assess the usability and efficiency of the authentication process, providing insights into how factors like device type and code length affect user experience and system performance.
- *True Positive Rate (TPR): How often do participants successfully authenticate in benign login sessions?* TPR reflects the percentage of benign authentication attempts that participants correctly approve, resulting in successful logins. It is calculated as:

$$TPR = \frac{\#\text{ of accepted benign sessions}}{\#\text{ of benign sessions}} \tag{1}$$

- *False Positive Rate (FPR): How often do participants incorrectly approve attack login sessions?* FPR measures the proportion of attack sessions that are incorrectly approved, leading to unauthorized access attempts being granted. It is calculated as:

$$FPR = \frac{\#\text{ of accepted attack sessions}}{\#\text{ of attack sessions}} \tag{2}$$

- *User Consistency Rate: How consistent are participants in their authentication responses across trials?* The consistency rate is defined as the proportion of correct responses for benign sessions (correct approvals) and correct rejections for attack sessions. By analyzing user consistency, we can identify patterns in user behavior and assess the reliability of the authentication process across different conditions. Furthermore, we conducted an accuracy analysis using a rating scale of 0 to 5, where a score of 5 indicated perfect accuracy.

When calculating TPR and FPR, we excluded push notifications that were either dropped by the network or deleted by participants, as no responses were received for these notifications. This exclusion ensures accuracy by focusing solely on notifications that were received and processed, reflecting the system's true performance.

To analyze the statistical significance of our results, we employed the Wilcoxon Signed-Rank Test (WSRT). This nonparametric test was chosen due to its robustness, as it does not assume normality—making it ideal for data that may not follow a normal distribution. The WSRT was applied at a 95% confidence level to compare performance metrics between groups (e.g., PC vs. phone, four characters vs. six characters). By comparing response times and accuracy metrics between PC and phone groups, as well as different code lengths, the WSRT helped us assess whether observed differences were statistically significant. This ensured that any detected differences were not merely due to random variation but indicated meaningful underlying patterns.

Furthermore, the Bonferroni correction was applied during post-analysis to account for multiple comparisons. This correction adjusts the significance level to mitigate the increased likelihood of making a Type I error when conducting multiple statistical tests. By using the Bonferroni correction, we reduced the chances of observing significant differences by chance alone. Employing this correction ensured that our statistical conclusions remained robust and reliable, even during multiple testing scenarios.

The effect size of the WSRT, an essential metric that transcends mere statistical significance to gauge the practical importance of the observed differences, was calculated as $r = Z/\sqrt{N}$, where Z is the value of the z-statistic and N is the number of observations on which Z is based. In this context, the effect size offers a clear perspective on the magnitude and relevance of the differences between groups. For interpretative clarity, an effect size of 0.1 might be considered small, 0.3 as medium, and 0.5 as large.

To further understand individual variability, we employed mixed-effects models that account for both fixed effects (e.g., device type, code length) and random effects (e.g., individual differences among participants). This approach allowed us to assess how various factors influence authentication performance while considering the inherent variability between users.

E. Study Protocol

We recruited 65 participants representing diverse educational backgrounds (demographic details are provided later in Section IV). At the beginning, each participant received a brief explanation of two-factor authentication (2FA) and push-based 2FA, ensuring familiarity with the core concepts of the Push-Compare-and-Confirm approach.

One researcher, serving as the examiner, guided each participant to a workstation equipped with a desktop PC and an Android smartphone. The desktop PC's browser was set to our study site, while the Android smartphone, set as the second-factor device, had our study app installed.

Participants were instructed to treat the desktop PC as if it were their personal computer for managing online accounts and to consider the Android smartphone as their personal phone for 2FA. They were informed that the login process simulated a real authentication scenario. This role-playing approach is commonly used in security and usability studies [21], [24], [25]

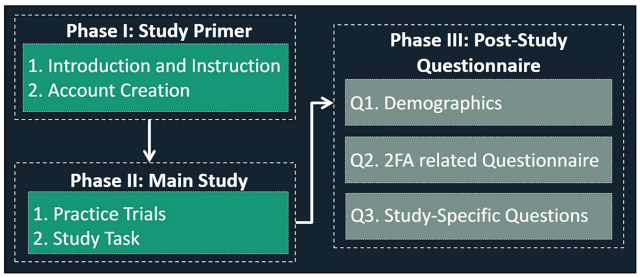


Fig. 5. Comprehensive overview of the study protocol flow for evaluating the Push-Compare-and-Confirm two-factor authentication method.

to enhance the authenticity of participant interactions without compromising actual account security.

To ensure all participants received equal information, study details were displayed on the welcome page of the study site. Participants were advised to review these details and encouraged to raise any questions about the study. They then created an account in our Push-2FA system, linked the Android phone as the second-factor device, and performed a total of 24 login attempts. Upon completing the login attempts, each participant answered a short survey capturing their impressions of the study, the Push-2FA method, and general perceptions of 2FA.

Each participant's session lasted approximately 25–30 minutes. The study received approval from our University's Institutional Review Board (IRB). Participation was voluntary, and standard ethical protocols were observed. Participants had the right to withdraw at any point, and all personal information (e.g., passwords) was permanently deleted at the study's conclusion to protect their privacy.

Phase I: Study Primer: This study started with a brief orientation, as illustrated in Fig. 5. Participants received an overview of two-factor authentication (2FA) concepts and the Push-Compare-and-Confirm method, focusing on its security objectives. They were informed that our goal was to observe their behaviors when using Push-Compare-and-Confirm under realistic login conditions. Participants were explicitly told that they could *Deny* any login notification if they detected suspicious activity. All introductory information and instructions were displayed on the study site welcome page.

1) *Introduction and Instructions:* Since some participants might be unfamiliar with 2FA and Push-Compare-and-Confirm, we provided a clear explanation of these methods and their security purposes. Participants were instructed to assume they were logging into personal services (e.g., a bank account) through our study system, aiming to complete the login attempt accurately. This role-playing approach, commonly adopted in security and usability research [21], [24], [25], helps simulate actual user behavior without involving real accounts.
2) *Account Creation:* Participants were prompted to create a new account within our Push-2FA system, choosing any username and password not already registered in our database. To replicate typical user practices, they could optionally use the browser's password manager and auto-fill features for convenience. We emphasized that only the second-factor authentication process, not the primary credentials, was the focus of our study. After participants set up their accounts, they linked the phone authentication app by signing in with the same credentials, similar to real-world two-factor authentication setups. This linkage process was performed only once, at the beginning of the study.

Phase II: Main Study: Once the participant's account was set up and the phone was linked as the second-factor device, the study proceeded with a practice session followed by the primary study tasks:

1) *Practice Trials:* To familiarize participants with the Push-Compare-and-Confirm system, they completed four initial login attempts—two in the PC setting and two in the phone setting. In each setting, participants tested one four characters code and one six characters code, ensuring they experienced both code lengths. These practice trials were strictly benign (i.e., the notification code matched the terminal's code), and any data collected was excluded from the final analysis. During these practice logins, participants could ask questions to clarify any aspects of the study and the Push-Compare-and-Confirm method.
2) *Study Task:* Before starting the main study phase, participants had the opportunity to raise any additional questions and confirm their understanding of Push-Compare-and-Confirm. They then performed twenty-four total login sessions: twelve in the PC setting (desktop PC as authentication terminal and the smartphone as second-factor device) and twelve in the phone setting (smartphone serving as both authentication terminal and second-factor device). Fig. 9 in the Appendix presents the hierarchical arrangement of these login sessions, providing a structured framework for evaluating Push-Compare-and-Confirm across different code lengths and device scenarios.

Phase III: Post-Study Questionnaire: After completing the main study, participants were asked to fill out a post-study questionnaire designed to gather additional information and feedback regarding their demographics, experience with two-factor authentication (2FA), and specific experiences with the Push-Compare-and-Confirm (Push-Compare-and-Confirm) method. We provide an overview of the main questions here. Refer to the Appendix for the complete list of questionnaires, 6.2.

The questionnaire comprised three main sections:

1) *Demographics and Background:* This section collected information on participants' age, gender, educational level, and self-assessed computer and security skills. The purpose was to understand the diversity of the participant pool and to contextualize their experiences with the Push-Compare-and-Confirm method.
2) *2FA-Related Questions:* Participants were queried about their familiarity with 2FA and Push-based 2FA. Key questions focused on the types of 2FA methods they have used, the services for which they employ 2FA (e.g., banking, email, social media), the frequency of their 2FA usage, and

the type of terminal they typically use for logins. These questions aimed to assess participants' prior experience with 2FA, which could influence their interaction with the Push-Compare-and-Confirm system.

3) *Study-Specific Questions:* This section explored participants' experiences during the study, gathering insights into their behaviors and preferences when using the Push-Compare-and-Confirm method. Main questions included:
   - How often did they verify/match the code on the login notification?
   - Which type of terminal did they prefer for logging in?
   - Which code length did they prefer?
   - How confident were they in the security of the Push-Compare-and-Confirm method?

   Participants were also encouraged to provide qualitative feedback on any challenges faced, moments of uncertainty, aspects they liked most about the Push-Compare-and-Confirm method, and suggestions for improvement.

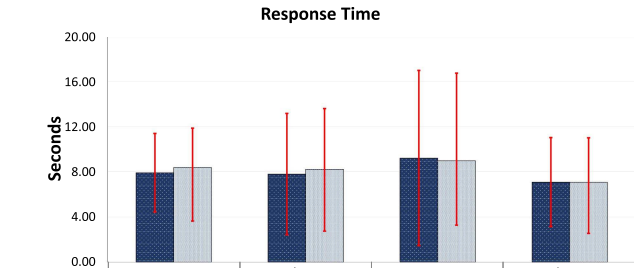
The questionnaire was designed to capture both quantitative data and qualitative insights, allowing for a comprehensive analysis of the usability and security perceptions of the Push-Compare-and-Confirm method from the users' perspectives. The combination of structured questions and open-ended responses was instrumental in identifying strengths and areas for enhancement within the system. Participants completed the questionnaire in approximately 5–10 minutes. Ethical considerations were carefully observed, ensuring confidentiality and anonymity of responses in accordance with guidelines for research involving human subjects.

IV. Analysis and Results

In this section, we present the quantitative and qualitative results derived from our evaluation of the Push-Compare-and-Confirm authentication scheme. The analysis is structured around four primary areas: Timing, Benign Sessions, Attack Sessions, and Post-Study Questionnaire Insights. We begin by examining the authentication response times across different device types and code lengths. Following this, we assess the system's accuracy through True Positive Rates (TPR) in benign scenarios and False Positive Rates (FPR) in attack scenarios. Lastly, we delve into the participants' feedback to contextualize our quantitative findings with their subjective experiences and perceptions. This comprehensive analysis provides a clear understanding of Push-Compare-and-Confirm's performance, highlighting both its strengths and areas for improvement.

A. Timing

To assess the timing of authentication using Push-Compare-and-Confirm, we measured the duration from when the server began verifying the first factor to when it received the response (Approve or Deny) from the participants, as explained in Section III. To maintain accuracy across devices, timestamps were logged on the server to prevent discrepancies caused by differences in the clocks of authentication terminals. The average response times for each setting are illustrated in Fig. 6.



	Benign PC	Benign Phone	Attack PC	Attack Phone
4 Characters	7.91	7.79	9.21	7.09
6 Characters	8.38	8.22	8.97	7.06

Fig. 6. Mean and standard deviation of response times (in seconds) across experimental conditions.

Our results showed that participants had similar response times for login notifications in both benign and attack sessions. Overall, the average response time (standard deviation) in the benign sessions across all settings was 8.08 seconds (4.80 seconds), while in the attack sessions, it was also 8.08 seconds (5.72 seconds).

Response Time in Benign Sessions: The results for benign sessions under PC and phone settings are shown in the first two blocks of Fig. 6. In the PC setting, participants took an average of 8.15 seconds (SD = 4.14) to respond to both four characters and six characters codes. Specifically, they spent 7.91 seconds (SD = 3.48) responding to four characters codes and 8.38 seconds (SD = 4.74) responding to six characters codes. As expected, participants took slightly more time with six characters codes. In the phone setting, participants took a similar amount of time to respond, with an average of 8.01 seconds (SD = 5.39). This breaks down to 7.79 seconds (SD = 5.37) for four characters codes and 8.22 seconds (SD = 5.47) for six characters codes.

Response Time in The Attack Sessions: In the attack sessions, participants had similar response times to the benign sessions. Since they only needed to match the initial characters to identify a mismatch, if the first characters did not match the one displayed on the authentication terminal, they could reject (deny) the login notification without matching the remaining characters. In the PC setting, participants took an average of 9.09 seconds (6.76) overall. Specifically, they spent 9.21 seconds (7.75) responding to four characters codes and 8.97 seconds (5.69 seconds) for six characters codes. In the phone setting, participants had an overall average response time of 7.08 seconds (4.21), with 7.09 seconds (3.94) for four characters codes and 7.06 seconds (4.51) for six characters codes.

Statistical Analysis of Response Times: We used the Wilcoxon Signed-Rank Test (WSRT) to analyze average response times across device types (PC vs. phone), code lengths (four characters vs. six characters), and session types (Benign vs. Attack), resulting in a total of 12 comparisons. These comparisons covered all pairwise combinations of the factors:

- *Device Type Comparisons (4 comparisons):* PC vs. phone under each combination of session type (Benign, Attack) and code length (four characters, six characters).

- *Code Length Comparisons (4 comparisons):* four characters vs. six characters codes under each combination of device type (PC, phone) and session type (Benign, Attack).
- *Session Type Comparisons (4 comparisons):* Benign vs. Attack sessions under each combination of device type (PC, phone) and code length (four characters, six characters).

At the 0.05 significance level, significant differences were found in three comparisons:

1) PC vs. phone in attack sessions with four characters codes ($p = 0.024$, $r = 0.29$).
2) PC vs. phone in attack sessions with six characters codes ($p = 0.012$, $r = 0.34$).
3) Attack vs. benign sessions on phone with six characters codes ($p = 0.031$, $r = 0.26$).

To control the Type I error due to multiple comparisons, we applied the Bonferroni correction by dividing the alpha level (0.05) by the number of comparisons (12), yielding an adjusted significance level of $\alpha = 0.0042$. After this correction, none of these differences remained statistically significant. These small to medium effect sizes suggest potential practical significance, indicating participants may respond differently under certain conditions. Overall, average response times are generally consistent across device types, code lengths, and session types in our study.

Mixed-Effects Model Analysis: Given the repeated-measures design of our study, where each participant interacted with both PC and phone devices under various conditions, we employed a mixed-effects model [26] to analyze the response times. This approach accounts for both fixed effects—**Device Type** (PC vs. phone), **Code Length** (four characters vs. six characters), and **Session Type** (Benign vs. Attack)—and random effects attributable to individual differences among participants. Including *Participant ID* as a random effect control for intra-subject correlations and variability, providing more accurate estimates of the fixed effects. The mixed-effects model was specified with *Average Response Time* as the dependent variable. The fixed effects included:

- *Device Type:* PC (reference category) vs. phone
- *Code Length:* four characters (reference category) vs. six characters
- *Session Type:* Attack (reference category) vs. Benign

The random effect was modeled with a random intercept for *Participant ID*.

Results: The mixed-effects model analysis yielded the following results:

- *Device Type (PC vs. phone):* A statistically significant effect was observed ($\beta = -1.078$, $SE = 0.376$, $z = -2.866$, $p = 0.004$). Participants responded significantly faster on phone compared to PC, with an average decrease in response time of approximately 1.078 seconds.
- *Code Length (four characters vs. six characters):* No significant effect was found ($\beta = 0.161$, $SE = 0.376$, $z = 0.428$, $p = 0.669$), indicating that code length did not significantly impact response times.

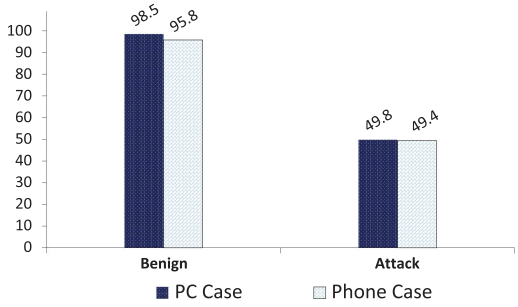


Fig. 7. Average of true positive rate (TPR) and false positive rate (FPR) of push-compare-and-confirm in PC and phone cases.

- *Session Type (Attack vs. Benign):* No significant effect was detected ($\beta = -0.006$, $SE = 0.376$, $z = -0.017$, $p = 0.986$), suggesting that response times were similar between attack and benign sessions.

The variance component for the random effect of *Participant ID* was estimated at 7.452, indicating substantial variability in response times attributable to individual differences. This underscores the importance of accounting for individual variability in the analysis. Overall, the mixed-effects model reveals that **Device Type** significantly influences response times, with participants responding faster on phone than on PC, while *Code Length* and *Session Type* do not have a significant impact.

B. Benign Sessions

In the benign sessions, we measured the True Positive Rate (*TPR*) to assess participants' success in correctly approving legitimate login attempts using Push-Compare-and-Confirm. The *TPR* represents the percentage of successful logins where participants correctly matched the unique code displayed on the login notification.

A total of over 750 responses were recorded in the benign sessions. Across all benign settings (PC and phone), the overall *TPR* was 97.12%, indicating a high level of success in matching the unique codes (see Fig. 7). This high *TPR* demonstrates the effectiveness of Push-Compare-and-Confirm in facilitating accurate authentication.

In the PC setting, participants achieved an overall *TPR* of 98.48%. Specifically, the *TPR* was 99.51% for the four characters codes and 97.55% for the six characters codes (see Fig. 8). The Wilcoxon Signed-Rank Test (WSRT) did not reveal a statistically significant difference between the four characters and six characters codes in the PC setting ($p = 0.102$), indicating consistent performance across different code lengths.

In the phone setting, the overall *TPR* was 95.76%. The *TPR* was 94.98% for the four characters codes and 96.81% for the six characters codes. The WSRT also did not indicate a statistically significant difference between the four characters and six characters codes in the phone setting ($p = 0.366$).

Comparing the overall *TPR* between the PC and phone settings revealed a statistically significant difference ($p = 0.041$), suggesting that device type impacted performance in benign sessions. Participants performed slightly better on the PC than

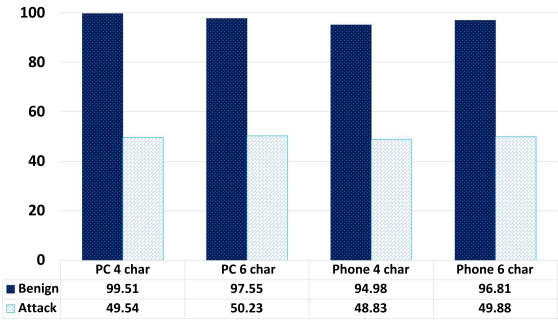


Fig. 8. Performance of participants with different settings in the benign and attack sessions.

on the phone. These results demonstrate that participants were highly successful in correctly approving legitimate login attempts using Push-Compare-and-Confirm, with high *TPR* across all settings. The significant difference between PC and phone settings suggests that participants found it slightly easier to use Push-Compare-and-Confirm on the PC, which may be due to factors such as screen size or interface layout.

C. Attack Sessions

In the attack sessions, we measured the False Positive Rate (*FPR*) to assess participants' ability to reject illegitimate login attempts correctly. The *FPR* represents the percentage of attack login attempts that were incorrectly approved by participants

A total of over 750 responses were recorded in the attack sessions. Across all attack settings, the overall *FPR* was 49.56%, indicating that participants incorrectly approved approximately half of the attack login attempts (see Fig. 7). This suggests that participants faced challenges in detecting and rejecting attack attempts.

In the PC setting, the *FPR* was 49.77%. Specifically, the *FPR* was 49.54% for the four characters codes and 50.23% for the six characters codes (see Fig. 8). The WSRT did not reveal a statistically significant difference between the four characters and six characters codes in the PC setting ($p = 0.950$), indicating similar performance across code lengths.

In the phone setting, the overall *FPR* was 49.36%. The *FPR* was 48.83% for the four characters codes and 49.88% for the six characters codes. Again, the WSRT did not indicate a statistically significant difference between the four characters and six characters codes in the phone setting ($p = 0.600$). Comparing the *FPR* between the PC and phone settings showed no statistically significant difference ($p = 0.352$), suggesting that device type did not impact participants' ability to detect and reject attack login attempts. These results indicate that participants had difficulty in identifying attack attempts, incorrectly approving approximately half of them regardless of device type or code length. This highlights a potential vulnerability in the Push-Compare-and-Confirm method, where users may not adequately verify the unique codes during authentication, leading to a high *FPR* in attack scenarios.

D. Post-Study Analysis

This section analyzes post-study data, covering participant demographics, 2FA practices, and user experiences with the Push2FA-Compare-and-Confirm method. We first review demographic and 2FA characteristics to contextualize participants' familiarity with security practices, followed by an analysis of study-specific responses, including preferences, challenges, and confidence levels. Detailed tables and visualizations can be found in Appendix 6.4, available online.

Demographics and Background: Our study included 65 participants with a diverse demographic background. There were more male participants (56.9%) than female participants (43.1%). The participants were largely young adults: 18–24 years old (24.6%), 25–34 (60%), 35–44 (12.3%), and 45 years and above (3.1%). Educational attainment among participants varied: 13.8% were high school graduates, 10.8% had some college credit but no degree, 46.2% held a bachelor's degree, 20% had earned a master's degree, and 9.2% held a doctorate degree. Participants self-assessed their computer skills as excellent (43.1%), good (32.3%), fair (21.5%), or poor (3.1%). Regarding security skills, they reported excellent (49.2%), good (18.5%), fair (23.1%), or poor (9.2%). This demographic information suggests that our participants were generally well-educated and possessed a good level of computer and security proficiency, making them suitable for our study on Push-Compare-and-Confirm.

2FA-Related Questionnaire: A significant majority (86.2%) of participants had used two-factor authentication (2FA) before, while 13.8% had not. Regarding the frequency of 2FA use, 60% of participants reported using it daily, 36.9% used it sometimes, and 3.1% indicated they never use it. Participants reported using various types of 2FA methods: text-based (SMS) and software token-based 2FA were equally popular at 53.8% each, push-based 2FA was used by 30.8%, and hardware token-based 2FA was the least common at 4.6%. These percentages do not sum to 100% because many participants have used multiple 2FA schemes.

Regarding the services where 2FA was used, 89.2% of participants mentioned using 2FA in banking services, 75.4% in email accounts, 70.8% in work-related accounts (e.g., university or company), and 60% in social media platforms like Facebook and X (formerly Twitter). These percentages also do not sum to 100% because many participants have used 2FA in multiple services. We asked the participants about the type of terminal they preferred to log into their accounts, and we found that 90.8% preferred using a phone, while 9.2% preferred using a desktop or laptop. This data indicates that our participants were familiar with 2FA and regularly engaged with it across various services, highlighting their relevance to our study.

Study-Specific Questionnaire: This section explores the participants' specific behaviors and preferences during our study.

We asked participants how often they verify or match the code before responding to the login notification. A total of 41.5% of participants said they always verify the code, 30.8% often verify it, 24.6% sometimes verify it, and 3.1% never verify the code. Regarding terminal preference during the study, responses were

varied: 30.8% found PCs (laptops or desktops) easier to use, 40% preferred phones, 27.7% felt that both phones and PCs were equally easy to use, 1.5% said both terminals were equally difficult to use.

Participants who preferred PCs cited ease of code verification and less annoyance compared to switching between apps on phones. Those who preferred phones emphasized quick access and verification, portability, and not needing to type codes. Comments included "easy and quick access and verification," "the phone accompanies you everywhere," and "I can get the notification easily and quickly, plus I don't need to type the codes." Some participants mentioned that phones are "easier to use," "easy and convenient," and provide a better user experience.

We then asked the participants about their preference for code length. A majority, 55.6%, preferred four-character codes, citing ease of memorization and quicker processing. Comments included "easier to remember," "not wasting time and easy to use," and "easy to match." One participant noted, "the shorter it is, the faster the login," emphasizing the convenience of shorter codes. In contrast, 20.6% preferred six-character codes due to perceptions of enhanced security and reduced susceptibility to guessing attacks. Participants who favored longer codes made comments such as "I prefer a 6-character code and I believe it is more secure than a 4-character code," and "more security," highlighting their trust in the added complexity of longer codes. Additionally, 20.6% felt that both four- and six-character codes were equally easy to use, while 3.2% did not express any preference between them. This lack of strong preference suggests that factors other than code length might influence their user experience.

Participants were also queried about any challenges they faced while using the Push-Compare-and-Confirm method. Over half (51.2%) reported no challenges. However, 20.9% found it difficult to compare codes, particularly when switching between applications on their phones. Another 18.6% felt unsure about whether to approve or reject notifications, with some expressing frustration over the need to switch between apps. Additionally, 2.3% mentioned that when they rejected a login attempt due to a code mismatch, they expected to receive feedback or a response confirming the rejection. Some participants noted that inconsistent codes or unclear prompts led to hesitation or confusion, emphasizing the need for more precise feedback when codes did not match. It is important to note that in the context of our study, a code mismatch indicates a potential attack login attempt and any rejection notification would be sent to the login terminal, which in this scenario would be the attacker's terminal. This lack of immediate feedback on their own device may have contributed to participants' confusion.

In terms of confidence in the security of the Push-Compare-and-Confirm system, a substantial majority of participants expressed high confidence, with 43.1% rating it as 5 (highly confident) and 29.2% as 4. Participants who felt confident often cited the certainty provided by the code verification process and the security of receiving the code on their personal devices. Comments included "being sure about the code and the app sent the verification code" and "it's the difficulty of obtaining the code because it's on your phone and no one can easily access your phone." Additionally, one participant noted, "When the code was different, the system does not log me in, and that felt more secure," reinforcing trust in the system's ability to prevent unauthorized access. A few participants had reservations or felt less confident. One simply responded "No" when asked if any aspects made them feel more secure or less secure. Another commented, "Even if I agreed, it would not have made it easy to access my account," indicating some uncertainty about the method's effectiveness. One participant stated that it was "easy to access," which could refer to the ease of use enhancing confidence or raising concerns about security. Overall, participants' confidence levels were influenced by the personalized code verification process and the control over access provided by their personal devices.

We also explored whether participants experienced moments of uncertainty when deciding to approve or reject notifications. A total of 41.9% indicated they felt unsure sometimes, 34.9% said this happened rarely, 18.6% stated they never felt unsure, and 4.7% mentioned they often felt unsure. Situations causing uncertainty included encountering codes that differed from expectations, leading to hesitation. Some participants reported being unsure about the correct code, prompting them to double-check or reject the notification. Others noted that unclear or inconsistent instructions could cause doubts about whether to approve or reject a notification. These uncertainties highlight areas where the Push-Compare-and-Confirm method could be improved to enhance user confidence and decision-making during the authentication process.

When asked what they liked most about the Push-Compare-and-Confirm method, 37.2% appreciated the ease of use, 32.6% highlighted the clarity of notifications, 25.6% mentioned the sense of security the system provided, and 4.7% valued that notifications were clear and straightforward. Participants suggested several improvements to enhance the Push-Compare-and-Confirm method: 25.6% recommended simplifying the process to make it more user-friendly, 16.3% wanted even clearer notifications to reduce confusion, and 14% suggested using shorter codes for quicker verification. Other proposals included making it easier to switch between applications or improving the overall design to enhance the user experience. Some participants also recommended increasing security by requiring users to manually enter codes instead of relying solely on push notifications, indicating a desire for more active user involvement in the authentication process.

Overall, these findings suggest that while the Push-Compare-and-Confirm method is generally well-received in terms of security and usability, there are notable areas for improvement. Enhancing the user experience on mobile devices, providing clearer instructions, and simplifying the authentication process could reduce uncertainty and increase user satisfaction. The participants' feedback underscores the importance of balancing robust security measures with usability to ensure that authentication methods are both effective and user-friendly. Addressing these concerns could lead to higher adoption rates and better security practices among users.

V. Discussion and Future Work

This study evaluated the Push-Compare-and-Confirm authentication scheme across various scenarios, including benign and attack sessions, device types (PC and phone), and code lengths (four characters and six characters). By analyzing user consistency, accuracy rates, response times, and post-study feedback, we identified key areas for improving authentication security and usability.

A. Key Findings

High Accuracy in Benign Sessions: Users demonstrated high accuracy in benign sessions across both devices and code lengths, averaging over 97%. Specifically, with four characters codes, accuracy was 99.51% on PC and 94.98% on phone; for six characters codes, it was 97.55% on PC and 96.81% on phone. This consistency in non-threatening scenarios suggests a stable, reliable user experience. The high accuracy rates indicate that users can effectively use Push-Compare-and-Confirm in typical login situations. The minimal difference between code lengths suggests that the complexity added by longer codes does not significantly hinder user performance in benign contexts. This finding aligns with prior research indicating that users can manage moderate increases in authentication complexity without substantial impacts on usability [27].

Challenges in Attack Detection: Accuracy dropped significantly in attack sessions, with users detecting only about 50% of attacks on both PC and phone, regardless of code length. This detection rate, close to chance, highlights the difficulty users face in distinguishing between legitimate and fraudulent requests. This substantial decline in accuracy during attack sessions suggests that users may not fully engage with the code verification process when it is most critical. Cognitive factors such as habituation to notifications, overconfidence in security measures, or the influence of multitasking may contribute to this oversight [28]. Users might quickly approve requests out of habit, especially if they perceive authentication prompts as routine tasks. Compared to prior work by Jubur et al. [5], where users achieved a 31% detection rate in similar scenarios, Push-Compare-and-Confirm showed improved, yet still insufficient, detection accuracy. This improvement may be attributed to the enhanced UI design in Push-Compare-and-Confirm, such as the use of bold fonts to highlight codes, which could aid in better recognition of authentication cues. However, the persistent low detection rates highlight the limitations of UI changes alone in addressing security behaviors.

User Consistency Across Trials: In benign sessions, users exhibited high response consistency, averaging 4.5 out of 5 on PC and 4.7 out of 5 on phone, indicating habitual and reliable authentication behavior. However, in attack sessions, consistency rates fell significantly, particularly on phone, with averages of 2.5 out of 5 on PC and 1.8 out of 5 on phone. This inconsistency suggests that users struggled to detect and reject illegitimate login attempts consistently. Factors such as smaller screen sizes, distractions, and the tendency to multitask on mobile devices may exacerbate this issue. The lower consistency on phone underscores the need to consider device-specific challenges in authentication design. These accuracy and consistency trends are illustrated in Appendix Fig. 12, available online, which compares performance across device types and session scenarios.

Impact of Code Length and Device Type: Extending code length from four characters to six characters had minimal impact on accuracy in both benign and attack sessions, suggesting that code length alone does not significantly influence user performance. Users may rely on pattern recognition or general impressions rather than detailed code verification, especially under time pressure [29]. In terms of device type, users achieved higher accuracy on PC than on phone in benign sessions (98.48% vs. 95.76%, $p = 0.041$). The ability to use a dual-screen setup—viewing information on both the PC screen and the phone screen side by side—likely facilitates more accurate code verification on PCs, as it allows participants to compare details more conveniently. In contrast, the similar attack detection accuracy across devices (49.77% on PC and 49.36% on phone, $p = 0.352$) indicates that the challenges in recognizing fraudulent requests are pervasive and not confined to a specific device type.

Response Times, Authentication Efficiency, and Individual Variability: Our analysis revealed that participants responded significantly faster on phone compared to PC, with the Mixed-Effects Model showing a response time difference of approximately 1.078 seconds ($\beta = -1.078$, $p = 0.004$). This suggests that mobile devices facilitate quicker authentication responses, possibly due to the convenience of accessibility or a more streamlined interface. However, this speed may come at the cost of reduced attention to detail, potentially contributing to lower accuracy in attack detection. Furthermore, code length and session type did not significantly impact response times, indicating that these factors are less critical in influencing user speed during authentication. The consistent response times across benign and attack sessions (both averaging approximately 8 seconds) suggest that users do not take additional time to scrutinize authentication requests in potentially risky situations. This behavior may increase vulnerability to attacks, as users treat all authentication prompts with the same level of attention, potentially overlooking critical discrepancies.

The average authentication time of approximately 8 seconds aligns with findings from previous studies on standard Push-2FA systems [21], underscoring the efficiency of Push-Compare-and-Confirm in maintaining swift authentication processes, which is critical for user acceptance. However, balancing this efficiency with enhanced security measures remains a challenge, as the maintained speed does not compensate for the observed decline in attack detection accuracy. Additionally, the Mixed-Effects Model revealed substantial variability in response times attributable to individual differences (variance = 7.452). Personal characteristics—such as familiarity with authentication processes, cognitive load, and multitasking behaviors—play a crucial role in how users interact with Push-Compare-and-Confirm. Recognizing this variability is essential for designing authentication systems that accommodate diverse user behaviors and capabilities.

Additional Metrics Analysis: Beyond basic accuracy rates, we examined performance metrics such as False Positive Rate (FPR), precision, and recall to gain a nuanced understanding of

Push-Compare-and-Confirm's effectiveness. In attack sessions, the FPR was notably high at approximately 49.56%, indicating that nearly half of the illegitimate login attempts were incorrectly approved. Precision and recall metrics were both around 50%, reflecting the system's limited ability to accurately distinguish between legitimate and fraudulent login attempts. These findings align with observed user behaviors, where participants struggled to engage fully with the code verification process during critical moments, possibly due to cognitive overload, habituation to notifications, or misplaced trust in authentication prompts.

User Confidence and Perception of Security: A substantial majority of participants expressed high confidence in the security of the Push-Compare-and-Confirm system, with 43.1% rating their confidence as 5 (highly confident) and 29.2% as 4. While high confidence is positive for user acceptance, it may contribute to complacency in security behaviors. Participants may overly trust the system and approve authentication requests without thorough verification, increasing susceptibility to attacks. Additionally, 41.9% of participants reported feeling unsure sometimes when deciding to approve or reject notifications, indicating that confidence does not necessarily equate to effective decision-making in security-critical tasks. Enhancing user education to balance confidence with attention is essential to ensure that trust does not lead to complacency.

User Preferences and Experience: The post-study survey revealed a strong preference for using phone for authentication due to convenience and accessibility. Users commented, "I prefer using the phone for verification because it accompanies you everywhere" and "I can get the notification easily and quickly, plus I don't need to type the codes." These insights highlight the importance of aligning authentication methods with user habits and preferences to enhance usability. However, some participants noted challenges with app-switching on phone, stating that "switching between apps on the phone can be a bit annoying when done frequently." This suggests that even preferred devices may present usability hurdles that need to be addressed in design [30].

Need for Improved User Education and Interface Design: The low accuracy in attack detection indicates that users may not perform code verification effectively when it is most critical. Post-study responses revealed a tendency to trust notifications, especially under time pressure or when multitasking. This underscores the need for user education to raise awareness about potential security threats and the importance of consistent authentication practices. Enhancing interface designs—especially on mobile devices—can improve security by making attack cues more noticeable. Incorporating more salient visual cues, such as color changes or animations, and requiring interactive elements that necessitate active engagement may improve users' ability to detect attacks. Prior research indicates that users often overlook subtle security indicators [28], [29], emphasizing the need for more prominent alerts.

B. Practical Implications

The findings have significant practical implications for the design of authentication systems. The preference for phone usage suggests that mobile-first design considerations are paramount. However, the lower consistency and higher response speeds on mobile devices indicate that special attention must be given to mobile UI design to mitigate security risks. Implementing mandatory interaction steps, such as requiring users to input or select matching codes actively, could reduce the quick approval of requests. Additionally, providing user education within the app through brief tutorials or contextual prompts may enhance users' security awareness and encourage more cautious behavior. Balancing security and usability is crucial. Overly complex security measures may deter users, while insufficient safeguards expose them to risks. Designing authentication processes that are intuitive and integrate seamlessly into users' workflows can improve compliance and security outcomes. Considering individual differences in user behavior, adaptive authentication mechanisms that tailor the experience based on user profiles may further enhance both security and usability.

C. Limitations and Future Work

Limitations: While our study provides valuable insights, several limitations should be acknowledged. First, the participant sample consisted predominantly of young, tech-savvy individuals, primarily students, which may limit the generalizability of our findings. Including a more diverse participant pool in future studies would enhance the applicability of the results. Second, the controlled lab environment may not accurately reflect real-world usage scenarios, as participants were aware of being part of a study, potentially influencing their behavior. Field studies or long-term deployments in naturalistic settings are recommended to observe authentic user interactions with Push-Compare-and-Confirm. Third, due to ethical considerations, we simulated attack sessions rather than testing on participants' real accounts, which may not fully capture the psychological factors present in real attack scenarios. Future research involving more realistic threat models could provide deeper insights into user behavior under genuine risk. Fourth, the study did not include open-ended questions in the post-study survey, which could have provided richer qualitative data. Incorporating open-ended responses in future research could uncover deeper motivations, perceptions, and challenges faced by users during authentication. Lastly, the sample size for certain subgroups (e.g., users who consistently failed to detect attacks) was relatively small, potentially affecting the statistical power of our findings and the ability to generalize conclusions about specific user behaviors.

Future Work: Building upon our findings, future research should explore avenues to enhance the Push-Compare-and-Confirm authentication scheme. Expanding participant demographics and conducting longitudinal field studies will help assess the scheme's effectiveness across diverse populations and over extended periods. Investigating interventions aimed at improving attack detection rates is crucial. Exploring enhanced interface designs with more salient security cues, dynamic visual indicators, or personalized messages may encourage users to pay closer attention during authentication. Incorporating user education components could also raise awareness about the

importance of code verification. Comparing Push-Compare-and-Confirm with emerging authentication methods, such as biometric authentication or risk-based systems, would provide valuable insights into its relative strengths and weaknesses. Understanding how Push-Compare-and-Confirm performs in relation to these methods can guide its integration into multi-factor authentication strategies. Given the significant individual variability observed in authentication performance, future research should consider adaptive authentication approaches that adjust security requirements based on user behavior, risk level, or contextual factors. Such adaptive systems may optimize the balance between security and usability, providing a tailored authentication experience that accommodates different user needs and capabilities. Furthermore, exploring the integration of Push-Compare-and-Confirm into smartwatch devices offers potential benefits, given their increasing popularity and convenience. However, challenges related to small screen sizes and user interaction need to be carefully addressed to prevent increased vulnerability to attacks. Future studies should focus on designing and evaluating Push-Compare-and-Confirm implementations on smartwatches, considering the unique usability and security implications of these devices. Overall, addressing these limitations and pursuing the proposed future research directions will contribute to the development of more secure and user-friendly authentication systems that effectively protect users while accommodating their needs and behaviors.

VI. Concluding Remarks

This study evaluated the Push-Compare-and-Confirm two-factor authentication (2FA) method, which enhances standard push-based 2FA by incorporating a code comparison step. Through 24 login trials across PC and phone devices with varying code lengths (four characters and six characters), we assessed Push-Compare-and-Confirm's usability and security.

Participants demonstrated high accuracy in benign login sessions, achieving True Positive Rates (*TPR*) exceeding 95% across all settings, indicating a reliable and user-friendly authentication experience. However, the method faced significant challenges in attack detection, with participants correctly identifying only about 50% of fraudulent login attempts. This False Negative Rate (FNR) suggests that while Push-Compare-and-Confirm improves upon standard push-based 2FA systems, vulnerabilities remain that require further mitigation. Potential factors contributing to this low detection rate include user complacency, interface design limitations, and the cognitive load during authentication.

Device preference analysis revealed that 40% of participants favored using the phone for authentication due to its convenience. However, challenges like app-switching and smaller screen sizes affected code verification accuracy on mobile platforms. Feedback from the study highlighted the importance of improved user education and more intuitive interface designs to enhance security effectiveness. Implementing these improvements can significantly lower the False Negative Rate, thereby strengthening Push-Compare-and-Confirm's overall security.

In conclusion, Push-Compare-and-Confirm demonstrates strong potential in providing a user-friendly authentication experience and offers notable improvements over traditional push-based 2FA in benign scenarios. However, addressing its current vulnerabilities is essential to enhance its security efficacy, making it a more robust and reliable 2FA solution for real-world applications.

Acknowledgment

The authors express sincere gratitude to the reviewers for their valuable feedback and insightful suggestions, which have significantly strengthened this paper. Additionally, the authors thank Asmaa Farea, Yasser ALshareef, and Asaad HEZAM for their generous support and efforts in recruiting participants, which were instrumental to the success of this study.

References

[1] DUO, "DUO push: Duo authentication," 2019. Accessed: Apr. 21, 2019. [Online]. Available: https://duo.com/product/trusted-users/two-factor-authentication/authentication-methods/duo-push

[2] A. Restaino, "One-tap sign-up and auto sign-in on websites," 2018. Accessed: Last Accessed: Jul. 01, 2018. [Online]. Available: https://developers.google.com/identity/one-tap/web/

[3] LastPass, "The only authenticator app you need," 2019. Accessed: Mar. 3, 2020. [Online]. Available: https://lastpass.com/auth/

[4] EwanD, "Enabling 2FA for MSA," 2017. Accessed: Last Accessed: May 11, 2019. [Online]. Available: https://bit.ly/2HihL9p

[5] M. Jubur, P. Shrestha, N. Saxena, and J. Prakash, "Bypassing push-based second factor and passwordless authentication with human-indistinguishable notifications," in *Proc. 2021 ACM Asia Conf. Comput. Commun. Secur.*, 2021, pp. 447–461.

[6] F. Home, "Internet statistics 2024: Facts and figures about online usage," 2024. Accessed: Sep. 30, 2024. [Online]. Available: https://www.forbes.com/home-improvement/internet/internet-statistics/

[7] T. Agency, "75 mobile surfing stats on internet traffic from mobile devices (updated)," 2024. Accessed: Sep. 30, 2024. [Online]. Available: https://thriveagency.com/news/75-mobile-surfing-stats-on-internet-traffic-from-mobile-devices-updated/

[8] BeyondTrust, "MFA fatigue attack," 2023. Accessed: Jan. 16, 2023. [Online]. Available: https://www.beyondtrust.com/resources/glossary/mfa-fatigue-attack

[9] Facebook, "Two-factor authentication for facebook now easier to set up," 2019. Accessed: May 10, 2019. [Online]. Available: https://bit.ly/2MpF3vP

[10] Google Inc., "Sign in faster with 2-step verification phone prompt," 2019. Accessed: Apr. 21, 2019. [Online]. Available: https://support.google.com/accounts/answer/7026266?co=GENIE.Platform%3DiOS hl=enoco=0

[11] Microsoft, "What is the Microsoft authenticator app?," 2020. Accessed: Jan. 21, 2021. [Online]. Available: shorturl.at/twBN0

[12] Google, "Firebase cloud messaging," 2021. Accessed: Last Accessed: Feb. 18, 2021. [Online]. Available: https://firebase.google.com/docs/cloud-messaging

[13] Apple, "Setting up a remote notification server," 2021. Accessed: Last Accessed: Feb. 18, 2021. [Online]. Available: https://developer.apple.com/documentation/usernotifications/setting_up_a_remote_notification_server

[14] J. M. Esparza, "Understanding the credential theft lifecycle," *Comput. Fraud Secur.*, vol. 2019, no. 2, pp. 6–9, 2019.

[15] A. Mirian, J. DeBlasio, S. Savage, G. M. Voelker, and K. Thomas, "Hack for hire: Exploring the emerging market for account hijacking," in *Proc. World Wide Web Conf.*, 2019, pp. 1279–1289.

[16] K. Gretzky, "Evilginx - Advanced phishing with two-factor authentication bypass," 2017. Accessed: Last Accessed: Apr. 19, 2019. [Online]. Available: https://breakdev.org/evilginx-advanced-phishing-with-two-factor-authentication-bypass/

[17] K. Gretzky, "Standalone man-in-the-middle attack framework used for phishing login credentials along with session cookies, allowing for the bypass of 2-factor authentication." 2018. Accessed: Feb. 21, 2021. [Online]. Available: https://github.com/kgretzky/evilginx2

[18] Microsoft, "Enable passwordless sign-in with the microsoft authenticator app," 2019. Accessed: Mar. 01, 2020. [Online]. Available: https://docs.microsoft.com/en-us/azure/active-directory/authentication/howto-authentication-passwordless-phone
[19] S. Jarecki, M. Jubur, H. Krawczyk, M. Shirvanian, and N. Saxena, "Two-factor password-authenticated key exchange with end-to-end password security," *Cryptol. ePrint Arch.*, Report 2018/033, 2018. [Online]. Available: https://eprint.iacr.org/2018/033
[20] J. Colnago et al., ""It's not actually that horrible": Exploring adoption of two-factor authentication at a university," in *Proc. 2018 CHI Conf. Hum. Factors Comput. Syst.*, ACM, 2018, Art. no. 456.
[21] K. Reese, T. Smith, J. Dutson, J. Armknecht, J. Cameron, and K. Seamons, "A usability study of five two-factor authentication methods," in *Proc. 15th Symp. Usable Privacy Secur.*, 2019, pp. 357–370.
[22] E. Uzun, K. Karvonen, and N. Asokan, "Usability analysis of secure pairing methods," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, Springer, 2007, pp. 307–324.
[23] Apache Friends, "XAMPP Apache+ MariaDB+ PHP+ Perl," 2019. Accessed: Oct. 9, 2020. [Online]. Available: https://www.apachefriends.org/index.html
[24] M. Shirvanian, S. Jareckiy, H. Krawczykz, and N. Saxena, "SPHINX: A password store that perfectly hides passwords from itself," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, IEEE, 2017, pp. 1094–1104.
[25] S. G. Lyastani, M. Schilling, M. Neumayr, M. Backes, and S. Bugiel, "Is FIDO2 the kingslayer of user authentication? A comparative usability study of FIDO2 passwordless authentication," in *Proc. 2020 IEEE Symp. Secur. Privacy (SP)*, IEEE, 2020, pp. 268–285.
[26] J. Pinheiro and D. Bates, *Mixed-Effects Models in S and S-PLUS*, 2nd ed. New York, NY, USA: Springer Science & Business Media, 2006, doi: 10.1007/b98882.
[27] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano, "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," in *Proc. 2012 IEEE Symp. Secur. Privacy*, IEEE, 2012, pp. 553–567.
[28] S. Egelman, L. F. Cranor, and J. Hong, "You've been warned: An empirical study of the effectiveness of web browser phishing warnings," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 2008, pp. 1065–1074.
[29] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor, "Crying wolf: An empirical study of SSL warning effectiveness," in *Proc. USENIX Secur. Symp.*, Montreal, Canada, 2009, pp. 399–416.
[30] K. J. Kim and S. S. Sundar, "Does screen size matter for smartphones? Utilitarian and hedonic effects of screen size on smartphone adoption," *Cyberpsychol., Behav. Social Netw.*, vol. 17, no. 7, pp. 466–473, 2014.

Mohammed Jubur received the Ph.D. degree from the University of Alabama at Birmingham (UAB), Biringham, AL, USA, in 2021. He is currently a Distinguished Scholar in web authentication and security. He is also an Assistant Professor with the Department of Computer and Network Engineering, Jazan University, Jazan, Saudi Arabia. His impactful research has been showcased at premier conferences such as EuroS&P'21, ASIACCS'21, Mobicom 2023, and CCS2024, as well as in esteemed journals like *ACM TOPS*'21. He is dedicated to advancing academic excellence and mentoring the next generation of scholars in web security and authentication with Jazan University.

Nitesh Saxena (Senior Member, IEEE) received the bachelor's degree in mathematics and computing from the Indian Institute of Technology, Kharagpur, India, the M.S. degree in computer science from the University of California, Santa Barbara, Santa Barbara, CA, USA, and the Ph.D. degree in information and computer science from the University of California, Irvine, CA, USA. He has been with the Faculty of UAB and NYU Tandon. He was also with Nokia Research Center, Finland, and INRIA Rhone-Alpes, France. He is currently a Full Professor with Computer Science and Engineering, the Associate Director with Global Cyber Research Institute (GCRI) and a Engineering Dean's Research Fellow with Texas A&M University, College Station, TX, USA, and the Founding Director with the Security and Privacy in Emerging Systems (SPIES) Lab. He has authored or coauthored more than 170 journal and conference papers, many at top-tier venues in Computer Science, including IEEE/ACM Transactions, ISOC NDSS, ACM CCS, IEEE S&P, IEEE EuroS&P, ACM WWW, ACM WiSec, ACM ACSAC, ACM CHI, ACM Ubicomp, IEEE Percom, and IEEE ICME. His research interests include imobile and IoT device security, the emerging field of user-centered security, "neuro security," the newest field of security research that he has pioneered, and the increasingly important domain intersecting artificial intelligence and security, where he works in the broad areas of computer and network security, and applied cryptography. His research has been externally supported by multiple grants from federal agencies, NSF, DoJ, DOD, DHS, and NSA, and by gifts/awards/donations from the industry, including Google (2 Google Faculty Research awards), Microsoft Research, Cisco, Comcast, Intel, Nokia, Research in Motion, MxD, and Polytec. He has won the Distinguished Paper Award at NDSS 2014 and Mark Weiser Best Paper runner up Award at Percom 2019. He was the Research Director for the Computer Science Department at the University of Alabama at Birmingham (UAB) from 2016 to18. On the educational/service and workforce development front, he is currently a Co-Principal Investigator for TAMU's CyberCorps program funded by NSF, and was the Director and Principal Investigator for the UAB's Scholarship for Service (SFS) CyberCorps program from 2017 to 2021, and an Architect and the Co-Director for UAB's M.S. Program in Cyber Security from 2013 to 2021 (ranked best among all MS cybersecurity programs by Fortune). He was also the Principal Architect and the Co-Director of the M.S. Program in Cyber-Security at the New York University's Tandon School of Engineering. He has instructed dozens of core fundamental courses in Computer Science, including Computer Security, Network Security, Modern Cryptography, and Discrete Structures. He has also directly advised and graduated over a hundred graduate (Ph.D./M.S.) and undergraduate students as well as several high school/middle school students. He was/is an Associate Editor for flagship security journals, such as *ACM Transactions on Privacy and Security*, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and Springer's I*nternational Journal of Information Security*. His work has received extensive media coverage, for example, at New York Times, NBC, MSN, CNN, Fox, Discovery, ABC, Bloomberg, MIT Tech Review, IEEE Spectrum, New Scientist, ZDNet, ACM TechNews, Yahoo! Finance, Communications of ACM, Yahoo News, CNBC, Slashdot, Guardian, Computer World, Science Daily, and Motherboard.

Faheem A. Reegu received the bachelor's and master's degrees in computer science from Kashmir University, Srinagar, India, in 2010 and 2013, respectively. He is currently working toward the Ph.D. degree with Advance Informatics Department, Razzak Faculty of Technology and Informatics, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia. He is a Lecturer with the Department of Computer Science, Jazan University, Jazan, Saudi Arabia.