Measuring and Preventing Certificate Misconfigurations in Enterprise WPA2/3 Networks

Rathan Appana
DistriNet, KU Leuven
Leuven, Belgium
rathan.appana@student.kuleuven.be

Mathy Vanhoef
DistriNet, KU Leuven
Leuven, Belgium
Mathy.Vanhoef@kuleuven.be

***Abstract*—Enterprise Wi-Fi supports per-user credentials and is widely used in larger organizations or companies. However, securely configuring clients is error-prone and may expose users to Evil Twin attacks. In this paper, we use a rogue WPA2/3 access point to measure client misconfiguration rates among university students and staff. Our results indicate that one in five devices do not properly validate the server certificate and would have been exposed to credential theft. Additionally, we found that most clients, more than 90%, transmitted their username in plaintext, enabling trivial tracking attacks even when using MAC address randomization. To determine the cause of these issues, we analyzed client user interfaces on five operating systems, and found usability shortcomings in several of their certificate validation prompts. As a defense, we propose using EAP-pwd, which provides mutual client–server authentication using only the user's password, eliminating certificates.**

***Index Terms*—WPA2, WPA3, Enterprise, Certificate Validation, Trust On First Use, Anonymous Identity**

I. Introduction

Modern Wi-Fi networks are widely used since they provide seamless accessibility and mobility, and as a result, most modern devices rely on these networks to connect to the Internet or internal company networks. These Wi-Fi networks fall into two categories: Wi-Fi Protected Access (WPA) Personal and WPA Enterprise networks. Personal networks use a secret pre-shared password among all users, which makes them unsuitable for large-scale or organizational use. In contrast, Enterprise networks use 802.1X for authentication, enabling more flexible user management, making such networks a better fit for larger organizations or educational institutions.

The downside of Enterprise networks is that they are more tedious to securely configure [1], [2]. In particular, in most real-world networks, validating the certificate of the Authentication Server (AS) is a challenging problem. If the user is unaware of the correct AS details, disables AS certificate validation, or accepts any certificate prompt through Trust On First Use (TOFU), they are typically left vulnerable [3], [4]. An attacker can take advantage of this by setting up an Evil Twin (ET) that impersonates the legitimate network. This can mislead the user into joining the fake network, giving the attacker a chance to steal credentials and intercept their traffic.

In this work, we perform an ET measurement experiment where we impersonate eduroam to measure how secure clients are configured in practice. We also analyzed how different Operating Systems (OSes) handle self-signed, expired, and misconfigured certificates in their client User Interface (UI). All experiments were designed in such a way as to avoid harm to users, as will be further covered in Section III-D.

During our experiments, we also measured how many clients transmit their real identity in plaintext, instead of using an anonymous outer identity. Rather surprisingly, this revealed that more than 90% of KU Leuven users transmit their identity in plaintext. Finally, we discuss possible defenses to automatically use an anonymous outer identity, and to avoid having to manually verify the server's identity.

To summarize, our main contributions are:

- We show that evil twin attacks against Enterprise WPA2 and WPA3 networks remain an unsolved problem.
- We highlight shortcomings in the UIs of several Enterprise Wi-Fi clients.
- We propose adopting an updated version of EAP-pwd to avoid the usage of certificates altogether.
- We propose a method that allows clients to automatically use an anonymous outer identity.

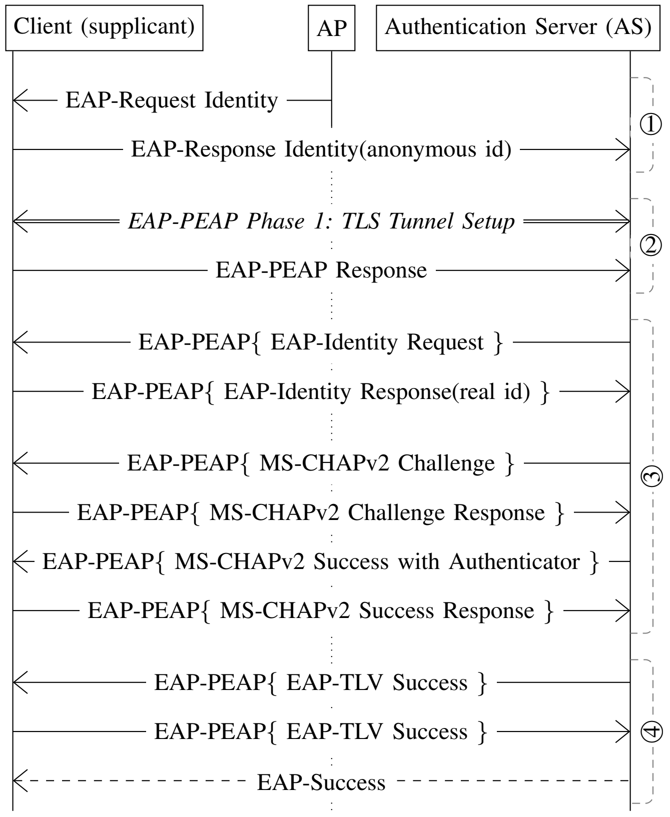
The remainder of the paper is structured as follows. Section II introduces essential background knowledge. Section III presents the setup of our measurement study, whose results are analyzed in Section IV. Section V presents our defenses and discusses results. Finally, we conclude in Section VI.

II. Background and Related Work

In this section, we introduce essential security building blocks of modern enterprise networks: authentication, certificate validation, and anonymous identities.

A. Enterprise WPA2/3 Authentication

In an Enterprise network, an 802.1X EAP-based handshake is performed to provide mutual authentication between clients, often also called supplicants in this context, and the network. As an example, Figure 1 shows an EAP handshake where Protected EAP (PEAP) with Microsoft Challenge Handshake Authentication Protocol version 2 (MS-CHAPv2) is used to authenticate the user based on their username and password. In stage ① of the handshake, the client sends a plaintext username to the AS. This identity can be an anonymous one, e.g., of the form `anonymous@realm`, and informs the AP which authentication server to use to authenticate the user.



Client (supplicant)
AP
Authentication Server (AS)
EAP-Request Identity
EAP-Response Identity(anonymous id)
EAP-PEAP Phase 1: TLS Tunnel Setup
EAP-PEAP Response
EAP-PEAP{ EAP-Identity Request }
EAP-PEAP{ EAP-Identity Response(real id) }
EAP-PEAP{ MS-CHAPv2 Challenge }
EAP-PEAP{ MS-CHAPv2 Challenge Response }
EAP-PEAP{ MS-CHAPv2 Success with Authenticator }
EAP-PEAP{ MS-CHAPv2 Success Response }
EAP-PEAP{ EAP-TLV Success }
EAP-PEAP{ EAP-TLV Success }
EAP-Success
①
②
③
④

Fig. 1: Example of a PEAP handshake with as Phase 2 authentication MS-CHAPv2. The notation EAP-PEAP$\{p\}$ represents that the packet p is encrypted using TLS.

When using PEAP, stage ② of the handshake will establish a TLS connection between the client and the Authentication Server (AS). During this phase, the client confirms the authentication server's authenticity by verifying its TLS certificate.

Next, in stage ③ of Figure 1, the Phase-2 authentication method is executed between the client and authentication server. These Phase-2 methods are executed within the TLS tunnel. This is done because some Phase-2 methods, such as MS-CHAPv2, have known vulnerabilities [5], [6], and executing them within a TLS tunnel mitigates these known vulnerabilities. In practice, the most common Phase-2 authentication method is MS-CHAPv2. Finally, in stage ④ of the EAP handshake, the authentication server will send a Success message to the client to indicate that authentication has been completed successfully.

B. Certificate Validation Challenges

Although AS certificate validation should prevent network impersonation, many supplicants are misconfigured or have unclear UIs, causing this verification to fail [7]. For instance, some devices allow insecure settings like "Do not validate" or prompt users to trust certificates without sufficient information. This enables ET attacks, where a rogue Access Point (AP) impersonates the enterprise network and captures authentication traffic, including MS-CHAPv2 challenge-response pairs that can be cracked offline [5], [6]. Advanced attackers may even proxy authentication traffic to the real network, creating a full man-in-the-middle scenario.

Proper documentation on how to securely configure clients is often also lacking. For instance, Bartoli et al. found many university Wi-Fi setup guides to be incomplete or misleading [1]. Additionally, users often are not aware of the correct Certificate Authority (CA) or AS domain during setup, preventing proper validation.

C. Anonymous Identities

To improve privacy, a common method is to use an anonymous identity in the outer Extensible Authentication Protocol (EAP) authentication (stage ① in Figure 1). This prevents exposing the user's real identity before the Transport Layer Security (TLS) tunnel is established. The real credentials are sent only after the server certificate is accepted. However, this feature requires explicit supplicant configuration and is not supported or enabled by default on our tested devices. Additionally, some networks do not permit the use of an anonymous identity as part of the outer authentication, requiring users to send their actual username in stage ① in Figure 1.

III. Methodology

This section explains how we performed an ET measurement against KU Leuven students and staff to determine whether clients remain misconfigured even today. The goal was to measure the number of clients with insecure configurations, i.e., whether they performed certificate validation and whether they used anonymous identities, without stealing any credentials or sensitive user data. All steps were carefully designed to follow ethical guidelines, and at no moment did we gain access to user credentials or other personal data.

A. Designing the Evil Twin Measurement

To perform the ET measurement, we need our rogue AP to look like a real "eduroam" network so that nearby clients will attempt to connect. Fortunately, we can easily extract the configuration of the eduroam network by inspecting beacon frames and connecting to it ourselves.

Based on the extracted configuration of the real network, the fake AP advertised WPA2-Enterprise with PEAP-MS-CHAPv2 as the authentication method, matching what is commonly used in real eduroam deployments [8]. The resulting hostapd configuration file of the rogue AP is shown in Listing 1.

B. Deployment location

To avoid disrupting any live networks, we selected a location where eduroam was not available but where many students were present. This ensured there were no real APs nearby with the same SSID and minimized disruption of real connections. This also meant we did not have to deauthenticate, or otherwise forcibly disconnect, clients that were already connected to the real eduroam network.

```
interface=wlan0
ssid=eduroam
hw_mode=g
channel=1

# WPA2-Enterprise settings
wpa=2
wpa_key_mgmt=WPA-EAP
rsn_pairwise=CCMP
ieee8021x=1
eap_server=1
eap_user_file=hostapd.eap_user

# Certificates
ca_cert=certs/ecc/ca.pem
server_cert=certs/ecc/server.pem
private_key=certs/ecc/server.key
private_key_passwd=whatever
```

Listing 1: Hostapd configuration for the Evil Twin AP.

C. *Attack Implementation*

To perform the ET measurement, we used a laptop running Ubuntu 22.04.4 LTS with a wireless interface capable of AP mode. Our rogue AP was created using a modified version of hostapd v2.11. Hostapd was configured to act as a WPA2-Enterprise AP with support for 802.1X and the EAP authentication method PEAP, using MS-CHAPv2 as the inner authentication method (see Listing 1).

To test whether clients properly verify the AS's TLS certificate, we generated custom server certificates using the FreeRADIUS bootstrap tools. This process created a fake CA and an AS certificate signed by it. Both certificates were self-signed and included basic attributes like a domain name and expiration date. This allowed our rogue AP to present a realistic certificate chain to clients, which is critical to test whether users' devices would validate or blindly accept it.

We modified the open-source hostapd daemon such that it would track the following statistics:

- The number of clients that attempted to connect. This will be used as a baseline to measure the percentage of vulnerable clients.
- The number of TLS tunnels successfully established, which indicates the client did not properly verify the server's TLS certificate.
- The number of supplicants who used an anonymous outer identity (recall stage ① in Figure 1).

Some clients might use auto-reconnection, which can make our data redundant. To avoid this, we hashed the MAC addresses using SHA1 and stored only the hash values. For every connection attempt, we checked if the hash was already present. If it was, we skipped counting it again. Most devices today use soft MAC randomization, where a new random MAC is generated for each new network. As a result, during auto-reconnection, most clients reuse the same randomized MAC, so our hash-based approach is valid even in the face of MAC address randomization. Only Windows offers an option to randomize the MAC address on every single connection, but this feature is not enabled by default. This makes our technique effective in preventing duplicate counts.

To detect anonymous outer identities used by KU Leuven members, we checked the format of the outer identities sent by the client. All real KU Leuven usernames start with a letter followed by numbers, e.g., `r0123456@kuleuven.be`. Any identity not following this pattern was considered an anonymous outer identity. To ensure the correctness of this approach, we only track clients from KU Leuven, i.e., clients whose outer identity ends with `kuleuven.be`.

The ET measurement flow is now as follows:

1) A supplicant attempts to connect to our rogue eduroam.
2) If the client ignores the invalid certificate, it will establish a TLS tunnel with our rogue AP.
3) Normally, the AS would then request the user's identity and challenge-response in MS-CHAPv2 (stage ③ in Figure 1). However, in our setup, we stopped the process *before* these frames were sent. In other words, we intentionally did not send the MS-CHAPv2 challenge to avoid capturing user credentials. This prevented any password or sensitive data from being obtained.

D. *Ethical Considerations*

During the experiment, the attack was conducted in a location where eduroam was not available, ensuring no interference with the legitimate network. We did not send MS-CHAPv2 challenges to clients after TLS tunnels were established, which prevented any password or credential extraction. Additionally, no user identities were stored, and only SHA1 hashes of MAC addresses were kept in memory for duplication. This setup allowed us to measure misconfigurations without exposing any sensitive user data or affecting normal network operations. While tools such as EAPHammer can also support automated ET attacks, we instead relied on a custom hostapd setup to implement these ethical safeguards.

IV. Results

This section is divided into two parts. The first part presents the results of the ET measurement. The second part summarizes the behavior of supplicant interfaces across different OSes observed during the experiment.

A. *Evil Twin Measurement Results*

We performed the ET measurement in two separate runs to first confirm that the measurement of certificate verification was working, before also measuring the usage of anonymous identities in the second run. The goal was to see how many devices would connect to our rogue network and proceed far enough in the authentication process to establish a TLS tunnel, which indicates they accepted our fake server certificate.

During the first run, 296 unique devices attempted to connect to our rogue AP (see Table I). Out of these, 53 devices (18%) established a TLS tunnel. This means nearly one in five clients ignored certificate validation and proceeded with the authentication. In the second run, out of 268 unique devices,

TABLE I: Results of the Evil Twin attack against KU Leuven students and staff.

	# Connections started	# TLS tunnels established	# Anonymous users
Experiment 1	296	53 (18%)	—
Experiment 2	268	51 (19%)	19 (7%)

TABLE II: Results of supplicant interface behavior.

OS	Baseline	Expired certificates	CA change
macOS (15.4.1)	Pop-up (with info)	Pop-up (with info)	Pop-up (with info)
iOS (18.5)	Pop-up (with info)	Pop-up (with info)	Pop-up (with info)
Android (15)	Pop-up (config)	Error screen, authentication fails	Pop-up (config), authentication fails
Windows (11)	Pop-up (without info)	Error screen, authentication fails	Pop-up (without info)
Linux (6.8.0)	No pop-up	No pop-up	No pop-up

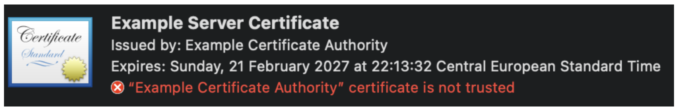
51 (19%) established a TLS tunnel. Additionally, we observed that 19 clients (7%) used an anonymous outer identity, e.g., `anonymous@kuleuven.be`, during the outer EAP stage.

Rather surprisingly, these results demonstrate that a significant percentage of clients remain misconfigured even today, and that they would have been vulnerable to credential theft in a real attack. Moreover, more than 90% of clients still do not use anonymous outer identities.

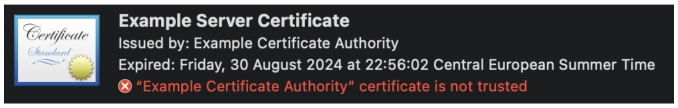
B. Supplicant Interface Study

To better understand why so many clients remain vulnerable to ET attacks, we studied the behavior of different OS interfaces. In particular, we first studied OS behavior when manually connecting to an Enterprise network for the first time. For this, we generated two sets of self-signed certificates, namely one set that was still valid, and another set that was already expired (2nd and 3rd columns in Table II). After the initial successful connection, we then also tested how different OSes react to changed CA certificates (4th column in Table II). In all tests, we did not preconfigure the OS with the expected certificate, but instead relied on Trust On First Use to establish trust in the certificate.

The behavior of different supplicants during the certificate validation process is summarized in Table II, pop-up (with info) means the certificate details were displayed to the user, allowing them to manually verify and accept or reject the certificate. Pop-up (configuration) means the supplicant presented a configuration interface where the user could choose validation options. Pop-up (without info) indicates a minimal warning was shown, but no certificate details were visible by default. No pop-up means the supplicant did not present any UI for certificate validation, effectively skipping certificate


Example Server Certificate
Issued by: Example Certificate Authority
Expires: Sunday, 21 February 2027 at 22:13:32 Central European Standard Time
"Example Certificate Authority" certificate is not trusted

(a) Valid certificate in macOS.


Example Server Certificate
Issued by: Example Certificate Authority
Expired: Friday, 30 August 2024 at 22:56:02 Central European Summer Time
"Example Certificate Authority" certificate is not trusted

(b) Expired certificate in macOS.

Fig. 2: Certificate prompts in macOS. Notice that the only difference between a valid and expired certificate is a single letter, i.e., "Expired" instead of "Expires".

pinning and validation. In all cases, if the user accepted the certificate, the connection succeeded unless failure is explicitly mentioned. More specifically, for each OS we observed the following:

macOS. We found that macOS displays certificate details on every change but shows expired and valid dates in the same font, where only the text "Expires" vs. "Expired" is different, making this easy to miss (see Figure 2). If the user accepts, authentication succeeds even with expired certificates.

iOS. iOS displays certificate details and highlights expired dates in red, drawing user attention. Accepting the certificate still allows authentication to succeed for expired certificates.

Android. Android offers more configuration options, and allows using system trusted certificates for validation, but it requires the user to specify the domain name of the AS. However, it does not provide support for selecting a specific CA from the trusted list. By default, Android fails authentication with expired certificates; the user cannot override this behavior.

Windows. Windows allows validating the AS certificate against all system trusted CAs or selecting one or more specific CAs from a trusted list. Despite this flexibility, it shows a basic warning with a hidden "show certificate details" option in the same font and color as the rest of the text, which can be ignored. It fails on expired certificates, and this behaviour cannot be overridden by the user.

Linux. Linux does not display any pop-up. Instead, the user must either provide a CA certificate to validate the incoming AS certificate or select the option "No CA certificate is required". Since users often do not have the CA certificate beforehand, they tend to select this latter option, which skips certificate validation entirely. Additionally, Linux does not provide any certificate pinning feature as other OSes provide.

V. Defenses and Discussion

In this section, we introduce novel defenses, discuss our results, and outline directions for future work.

A. Defense: Pre-Filled Anonymous Identity

In our experiments, more than 90% of users transmitted their identity in plaintext. To remedy this aspect, we propose that clients automatically fill in a default anonymous outer identity in their connection dialogue settings. In particular, if the user's real identity is of the form `username@realm`, then the supplicant can automatically suggest `anonymous@realm` as the plaintext outer identity, which the user can still optionally override if desired.

Here, it is essential that the original realm, which is often a domain name in practice, stays the same. This is because networks such as eduroam use the realm, e.g., `kuleuven.be`, to identify the authentication server to use.

B. Defense: Securely Verifying the Authentication Server

To seamlessly confirm the server's identity when connecting to a network for the first time, we propose using EAP Password (EAP-pwd) as inner authentication [3], [9]. We pick EAP-pwd because it is a password-authenticated key exchange that provides mutual authentication between the client and server. This approach prevents an ET from succeeding, as a rogue AP cannot complete the handshake without knowing the shared user password [9].

Moreover, even if the TLS tunnel is compromised when the client connects for the first time, the exchanged EAP-pwd packets do not reveal any information about the user's credentials. In particular, EAP-pwd is resistant to offline attacks, as it does not expose challenge-response pairs that can be cracked later. The downside of EAP-pwd is that it is difficult to implement without side-channel leaks [10]–[12]. Nevertheless, a draft proposal exists to strengthen its design to be more resistant to side-channel leaks [13], and we consider it an interesting future work to further develop and implement this improved version of EAP-pwd.

The main downside of EAP-pwd is that the user's identity must always be sent in plaintext. We consider it interesting future work to investigate techniques to better protect the user's identity when using EAP-pwd.

Another possible mitigation is the use of Configuration Assistant Tool (CAT), which enforces server certificate validation at the supplicant level by configuring the CA certificate and thus can reduce misconfigurations. However, CAT requires Internet access at first setup, and in practice, many clients bypass it to connect quickly.

C. Discussion: Enterprise Security Remains Challenging

The results of our ET measurement experiments highlight weaknesses in client configurations, showing that securely configuring Enterprise networks remains a challenging problem in practice. Nearly one in five devices connected to our rogue eduroam AP proceeded to establish a TLS tunnel, showing that many supplicants were either misconfigured or lacked proper certificate validation.

The problem of clients not verifying the server's certificate is worsened by the choice of inner authentication method. In particular, PEAP is widely used in enterprise networks, but its common combination with MS-CHAPv2 as inner authentication is problematic. In particular, an attacker capturing challenge-response MS-CHAPv2 pairs can perform an offline brute-force attack to recover user credentials [5], [6]. This makes the combination of misconfigured supplicants and weak inner authentication a significant security risk.

D. Future Work

Interesting future work is testing EAP-pwd to evaluate its usability and support. Further studies should assess timing attacks and optionally explore quantum-resistant alternatives to ensure this authentication mechanism remains future-proof [9]. Additionally, solutions are needed to also protect the user's identity when using EAP-pwd. Finally, adding TOFU mechanism on Linux, informed by the strengths and weaknesses observed in other OSes, would help result in a more secure and user-friendly supplicant.

VI. Conclusion

We showed that even modern WPA2 and WPA3 Enterprise clients remain vulnerable to ET attacks, where one in five clients unwittingly connected to the attacker's rogue network. Based on our analysis of UIs across five different OSes, especially Linux and older Android versions are insecure. To remedy the observed attacks, we propose using an updated version of EAP-pwd to verify the AS's authenticity based on the user's password.

Acknowledgment

This research is partially funded by the Research Fund KU Leuven and the Cybersecurity Research Programme Flanders.

References

[1] A. Bartoli, E. Medvet, A. De Lorenzo, and F. Tarlao, "(in)secure configuration practices of WPA2 enterprise supplicants," in *13th International Conference on Availability, Reliability and Security*, 2018.

[2] T. Perković, A. Dagelić, M. Bugarić, and M. Čagalj, "On wpa2-enterprise privacy in high education and science," *Security and communication networks*, no. 1, 2020.

[3] N. Asokan, V. Niemi, and K. Nyberg, "Man-in-the-middle in tunnelled authentication protocols," in *International Workshop on Security Protocols*. Springer, 2003, pp. 28–41.

[4] A. Cassola, W. K. Robertson, E. Kirda, and G. Noubir, "A practical, targeted, and stealthy attack against WPA enterprise authentication." in *NDSS*, 2013.

[5] M. Marlinspike, D. Hulton, and M. Ray, "Defeating PPTP VPNs and WPA2 enterprise with MS-CHAPv2," *Defcon, July*, 2012.

[6] B. Schneier, Mudge, and D. Wagner, "Cryptanalysis of microsoft's PPTP authentication extensions (MS-CHAPv2)," in *International Exhibition and Congress on Network Security*. Springer, 1999, pp. 192–203.

[7] I. Palamà, A. Amici, F. Gringoli, and G. Bianchi, ""careful with that roam, edu": experimental analysis of Eduroam credential stealing attacks," in *WONS*. IEEE, 2022.

[8] K. Wang, Y. Zheng, Q. Zhang, G. Bai, M. Qin, D. Zhang, and J. S. Dong, "Assessing certificate validation user interfaces of WPA supplicants," in *ACM WiSec*, 2022.

[9] G. Zorn and D. Harkins, "Extensible Authentication Protocol (EAP) Authentication Using Only a Password," RFC 5931, 2010. [Online]. Available: https://www.rfc-editor.org/info/rfc5931

[10] M. Vanhoef and E. Ronen, "Dragonblood: Analyzing the dragonfly handshake of WPA3 and EAP-pwd," in *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020.

[11] T. Van Goethem, C. Pöpper, W. Joosen, and M. Vanhoef, "Timeless timing attacks: Exploiting concurrency to leak secrets over remote connections," in *USENIX Security*, 2020.

[12] D. de Almeida Braga, P.-A. Fouque, and M. Sabt, "Dragonblood is still leaking: Practical cache-based side-channel in the wild," in *ACSAC*, 2020.

[13] D. Harkins, "Improved extensible authentication protocol using only a password," Internet Engineering Task Force, Internet-Draft draft-harkins-eap-pwd-prime-00, Jul. 2019, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-harkins-eap-pwd-prime/00/