A Practical Investigation of Identity Theft Vulnerabilities in Eduroam

Sebastian Brenza
Horst-Görtz Institute for IT-Security
Ruhr-University Bochum
Bochum, Germany
sebastian.brenza@rub.de

Andre Pawlowski
Horst-Görtz Institute for IT-Security
Ruhr-University Bochum
Bochum, Germany
andre.pawlowski@rub.de

Christina Pöpper
Horst-Görtz Institute for IT-Security
Ruhr-University Bochum
Bochum, Germany
christina.poepper@rub.de

ABSTRACT

Eduroam offers secure access to the Internet at participating institutions, using authentication via IEEE 802.1X and secure forwarding of authentication data to the authentication server of the user's institution. Due to erroneous configuration manuals and a lack of knowledge on the user side, though, a big share of client devices lack the required root CA certificate to authenticate the Eduroam network, yet still being able to access the network. Moreover, deficient software implementations on client devices prevent users from the secure execution of the authentication process.

In this paper, we present an attack that exploits this fact and uses the default behavior of wireless devices in order to capture authentication data. This MITM attack is performed in real-time. It is achieved using a modified version of hostapd, which exploits a compatibility setting of the widely used supplicant software wpa_supplicant. It enables an attacker to authenticate users in EAP-TTLS/PAP and in EAP-TTLS/MS-CHAPv2 without the necessity of cracking the user password hash on the fly and thus without inducing suspicious delays. In a practical study with several hundred users we could show that more than half of the tested devices were vulnerable to the attack. Based on the results of the study, we propose countermeasures to prevent the attack and minimize the amount of vulnerable devices.

Keywords

Network Security; WPA-Enterprise; Eduroam Authentication; EAP; MS-CHAPv2

1. INTRODUCTION

Access to scientific and educational resources is important in research and education. With the start of the Eduroam service at research and educational institutes, Internet access at participating institutes has been greatly simplified: Eduroam enables users to roam between participating institutions and authenticate with the login data of their home institutes which is securely forwarded for verification via a network of authentication servers to the home institution.

With their increased popularity in the last couple of years, many networked mobile devices have been configured for Eduroam usage by their users, including laptops, tablets, and smartphones. One central point to the security of the Eduroam usage is the proper setup of a root CA certificate to verify the authenticity of the network. However, secure authentication is not given, if deficient software implementations do not check the validity of the offered certificate correctly or client devices are wrongly configured with missing CA certificates. If the offered certificate is not checked correctly, client devices cannot verify the authenticity of the network, which opens the door for connecting to rogue access points, also called Evil Twins [12], and man-in-the-middle (MITM) attacks. This is an issue that also seems to be known in corporate networks [33].

Institutions participating in Eduroam are supposed to provide sufficient configuration instructions for their end users to allow for the authentic identification of access points at all times. Unfortunately, at a number of institutions participating in Eduroam, the configuration manuals for users are incomplete or have been so in the past. For example, at our university the configuration manuals for Android, Mac OS and iOS devices were missing the instructions for setting up the CA certificate. Fixing this simple problem of incomplete configuration manuals does not prevent the problem from prevailing in reality even months later, as we show by a practical study involving several hundred participants.

In this paper, we outline a number of problems regarding the authentication structure and processes of the Eduroam network. Although the Extensible Authentication Protocol (EAP) authentication methods used in Eduroam and Enterprise networks, in particular EAP-TTLS and PEAP, are well-defined and have been thoroughly examined for flaws by security experts, imprecisions in the implementations may lead to severe security vulnerabilities. We present a novel stealthy attack targeting client devices. The attack is based on real-world software and configuration deficiencies. In more details, we use a modified version of the software access point hostapd [20] that exploits the network's trust structure and enables an attacker to authenticate users on the fly without owning the required username and password

WiSec'15, June 22 – 26, 2015, New York, NY, USA
 ACM 978-1-4503-3623-9/15/06 ...$15.00
DOI: http://dx.doi.org/10.1145/2766498.2766512
.

data. This makes it possible to capture authentication data using an evil-twin access point and to gain control over the network traffic of the victim.

We investigate the applicability of the attack in a real-world setting and we identify, classify, and evaluate possible countermeasure – both on the side of the client devices as on the side of the infrastructure and service provider.

In summary, this paper makes the following contributions:

1. We present a novel attack on a widely used EAP-TTLS/MS-CHAPv2 implementation that allows to bypass the inner MS-CHAPv2 authentication on devices with deficient certificate validation.

2. We investigate the vulnerability of different Android- and Mac OS/iOS-based devices and tablets to this attack and show that they are all vulnerable (in some cases depending on their configuration and user interaction).

3. We report results on a practical study of this vulnerability with about 350 users and more than 500 devices. Of these tested devices, a share of 52 % turned out to be vulnerable to our attack.

4. We propose multiple client- and infrastructure-side countermeasures and evaluate their effectiveness against the presented attack. To minimize the share of wrongly configured devices, we show an automated way to check device configurations that can be deployed by institutions offering Eduroam services.

The rest of this paper is structured as follows: First, the technical background for the attack is presented in Section 2. Prerequisites and the attacker model are introduced in Section 3. In Section 4, the approach and characteristics of the attack are described, followed by details on the practical study in Section 5. Countermeasures are discussed in depth in Section 6. Finally, related work is presented in Section 7 and we conclude the paper in Section 8.

2. PRELIMINARIES

This section presents the technical background necessary for the attack described later in this paper. It introduces authentication in Eduroam and IEEE 802.1X along with the targeted tunneled authentication protocols as well as PAP and the MS-CHAPv2 authentication protocol.

2.1 Authentication in Eduroam

Authentication in Eduroam is carried out following the IEEE 802.1X standard for port-based network access control [17]. The system roles *Supplicant*, *Authenticator*, and *Authentication Server* (AS) correspond in Eduroam to the user's client device, the access point (AP), and the RADIUS authentication server of the user's home institution [30]. In order to enable users to gain network access using the same login data and network configuration at each participating institution, a hierarchical layer of RADIUS AS is introduced. The AS are organized on organizational, national, and international level to determine user affiliation. Eduroam provides mutual authentication either via X.509 client and server certificates or by tunneled authentication methods. This work focuses on tunneled authentication, which is carried out in two phases: In phase 1, the AS is either authenticated towards the user or both are mutually authenticated; in phase 2 the user subsequently authenticates towards the AS. The authentication is executed as follows (as shown in Figure 1):

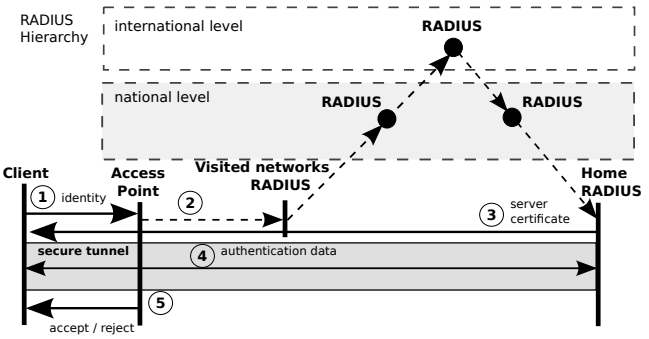

Figure 1: Authentication Process in Eduroam

Phase 1 (Outer Authentication):

1. When a user requests network access, the user's identity is queried by the AP. The user identity is provided to the AP in the format: `user@institution.tld`, where `tld` is the top-level domain.

2. The AP forwards the user identity to the home AS of the network, which checks its responsibility for the user's institution and top-level domain. In case the user requested access to a foreign network, the identity is proxied to the next RADIUS AS on national or international level—based on the user's identity—until the user's home AS is found.

3. The user's home AS checks if the identity is valid, starts tunnel establishment, and answers with its server certificate. The user then validates the server certificate and a secure tunnel is established between the user and the AS.

Phase 2 (Inner Authentication):

4. After the secure tunnel establishment, inner authentication is executed inside the tunnel between client and AS, using credentials of the user's home institution.

5. The user's home AS validates the login data and passes the result to the AP, which subsequently grants or refuses network access.

Due to privacy reasons it is encouraged to provide an anonymous outer identity in phase 1 and the actual/private identity inside the tunnel in phase 2 [30].

2.2 802.1X Protocols

During the authentication process in IEEE 802.1X, authentication and communication between the client device—also called *peer*—, the AP, and the AS are carried out using the Extensible Authentication Protocol (EAP).

EAP holds the data of the used authentication method, called *EAP method*, and is based on an exchange of EAP Request (sent by the authenticator) and EAP Response messages (sent by the client). In a typical EAP conversation

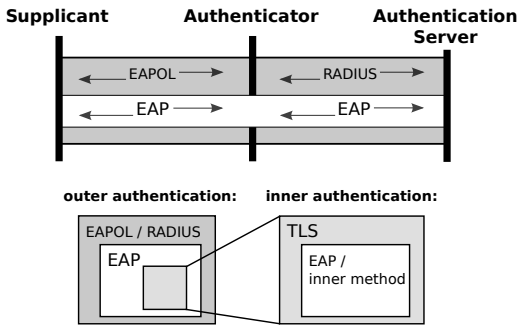

Figure 2: 802.1X Communication and Protocol Encapsulation

the authenticator first queries the identity with an EAP Request: Identity and concludes with an EAP Success or EAP Failure, depending on the authentication result [5].

EAP was designed to run within the point-to-point protocol (PPP). In order to operate in a LAN environment, like Ethernet or IEEE 802.11, EAP data is encapsulated between client and AP using EAP over LAN (EAPOL) as carrier protocol [17]. Prior to forwarding it to the AS, which implements the authentication method, the AP decapsulates the EAP data and encapsulates it using RADIUS as carrier protocol, which is the designated authentication protocol in Eduroam [30]. The communication between components in 802.1X, as well as the protocol encapsulation, are depicted in Figure 2.

Because the tunneled authentication methods used in Eduroam rely on Transport Layer Security (TLS) as secure tunnel, EAP packets also encapsulate the TLS records, which hold the inner authentication data in form of an EAP method or a different authentication method. In Eduroam basically every EAP-compatible authentication method that fulfills the Eduroam service definition can be used [30].

EAP Tunneled Authentication Methods

EAP-TTLS and PEAP are EAP methods for tunneled authentication that rely on TLS to provide mutual or server authentication and protection against man-in-the-middle attacks. The execution of EAP-TTLS and PEAP authentication are depicted in Figures 3 and 4.

The authentication methods consist of two phases, which start after an initial identity request: A handshake phase (step 1 in Figure 3 and 4), which establishes the TLS tunnel and is equivalent to Phase 1 in Section 2.1 and a data phase (step 2 in Figure 3 and 4), where inner authentication is executed, equivalent to Phase 2 in Section 2.1. As part of the TLS handshake, the AS certificate chain is provided, which is validated by the client. After successful authentication key material is distributed to the peer.

EAP-TTLS supports a selection of inner authentication methods, in particular the point-to-point protocols PAP and MS-CHAPv2, which are introduced below. The TLS handshake data is also used to implicitly generate challenge material for the inner authentication methods invoked in EAP-TTLS [14].

The PEAP version used in Eduroam is Microsoft's implementation of PEAP version 0, which only supports the EAP method EAP-MS-CHAPv2 as inner authentication method for non-certificate-based authentication [6][18].

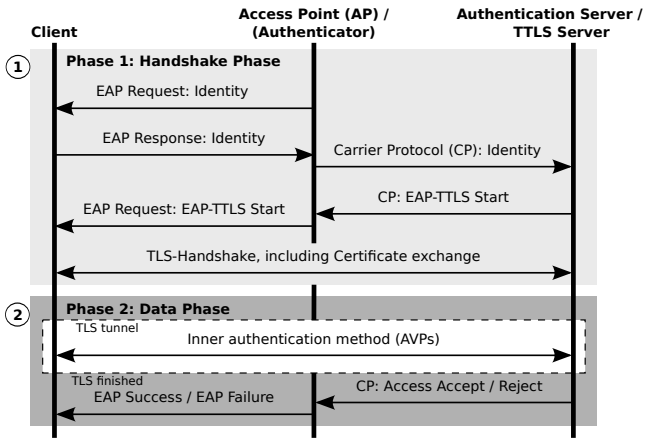

Figure 3: Sample EAP-TTLS Authentication Process

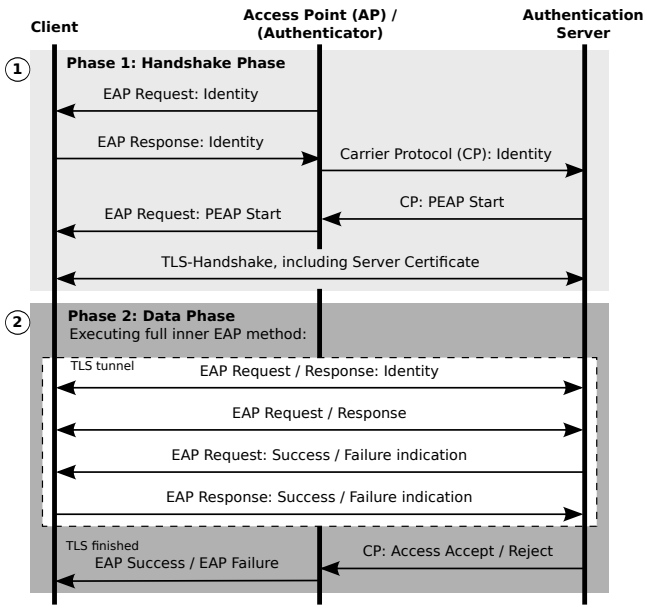

Figure 4: Sample PEAP Authentication Process

Because they are important for the attack presented later in this paper, a description of the used inner authentication methods follows.

PAP

In the password authentication protocol (PAP), the client authenticates against an AS using a combination of username and password. The client repeatedly submits an authentication request containing user identity and password until a response from the AS is received. The AS validates the transmitted data upon reception and answers either with an Access Accept or Access Reject [19].

The authentication data is submitted in plaintext without any form of protection and can thus be recovered by eavesdropping when not protected by other measures, such as a secure TLS [13] tunnel.

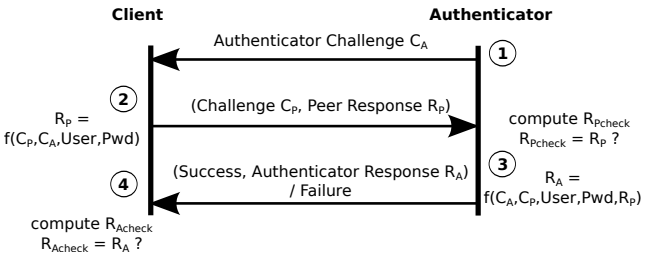

Figure 5: MSCHAPv2 Authentication Process

MS-CHAPv2

The Microsoft Challenge-Handshake Authentication Protocol Version 2 (MS-CHAPv2) is a challenge-response authentication protocol. MS-CHAPv2 extends MS-CHAP [36] and was designed to provide mutual authentication of authenticator and peer. The detailed authentication process is depicted in Figure 5 and described as follows [35]:

1. The authenticator chooses a random 16-octet long authenticator challenge C_A and sends it to the peer.
2. Upon reception, the peer chooses a C_P analogous to C_A and computes the 24-octet long Response R_P—also called NT-Response—as

$$R_P = \text{ChallengeResponse}(CHash, \text{MD4}(PW)), \quad (1)$$

   where

   - PW is the user's password
   - $CHash = \text{trunc}[0..7](\text{SHA1}(C_P||C_A||UserName))$
   - trunc[0..7] truncates the SHA1 digest to the first 8 octets
   - ChallengeResponse($\cdot$) provides DES encryption

   and sends $C_P||R_P$ to the authenticator as Peer Response.
3. The authenticator computes R_{Pcheck} and compares it to the received R_P. If $R_P = R_{Pcheck}$, the authenticator answers with the Authenticator Response

$$R_A = \text{SHA1}(D||CHash||M2), \quad (2)$$

   where

   - $D = \text{SHA1}(\text{MD4}(\text{MD4}(Password))||R_P||M1)$
   - $M1$ and $M2$ are "magic constants" specifically defined for response generation

4. The peer verifies the Authenticator Response and terminates the connection if $R_{Acheck} \neq R_A$, otherwise it continues.

The security of MS-CHAPv2 was thoroughly analysed in the past [25], pointing out its design flaws. In 2012 the complexity of the protocol was greatly reduced, making brute-force attacks feasible [22]. Nevertheless, MS-CHAPv2 is still used as inner authentication method in tunneled authentication protocols such as EAP-TTLS or PEAPv0, which are being used in Eduroam and enterprise networks.

3. SYSTEM AND ATTACKER MODEL

In this section, we present our assumptions about the network, the victim and his client device, as well as the capabilities of an attacker.

3.1 System Prerequisites

We assume that the following prerequisites are met regarding the configuration of the wireless network and the victims' clients:

(a) The wireless network of the victims uses WPA(2)-Enterprise for authentication and accepts the widely used methods EAP-TTLS/PAP, EAP-TTLS/ MS-CHAPv2, or PEAP/MS-CHAPv2 for authentication.

(b) The client has saved the target network in its known network list, e. g., due to prior connections, thus having configured an authentication method accepted by the network.

(c) The client connects to the wireless network AP with the highest signal strength while roaming. It also recognizes already known enterprise networks solely by their broadcasted SSID and automatically connects when a known network is in range. Note that this is the common default behavior on many devices.

(d) At least one of the following certificate prerequisites is fulfilled:

1. The client does not check the CN (common name) string of the offered certificate, thus lacking a validation of the server name. As shown in [3], client devices with such a deficient certificate validation are Android, Mac OS and iOS devices, unless the Apple devices are configured by installing a configuration profile.
2. The client has an insufficient device configuration in which the CA of the network is not setup. Thus the client does not validate the certificate chain during the handshake phase of the authentication process and accepts an arbitrarily offered certificate. Our practical study later in this paper (Sec. 5) demonstrates on a large number of devices that this case is not uncommon.

3.2 Attacker Model

Our assumptions about the capabilities of the attacker are:

- The attacker is able to communicate with the victim and the target network over the wireless channel. Also she is able to capture and modify forwarded network traffic using appropriate software, e. g., sslstrip [23].
- The attacker is able set up an own network AP with a signal strength higher than the signal strength of the target network.
- The attacker is able to forward network traffic to a valid network AP and set up an own DHCP-server.
- The attacker is able to gain a valid signed certificate with an arbitrary CN string by one of the CAs which has the same top-level authority as the Eduroam RADIUS server certificate.

Note that in the case of Eduroam, usually one top-level CA is used for the whole infrastructure of a country. This means when an attacker is able to get a certificate signed by this top-level CA, she is able to attack every client that

uses the Eduroam infrastructure of that country. For example, some universities offer free signed server certificates within their domain namespace (which uses the Eduroam top-level CA) with the only restriction that an ID card has to be shown. This service creates a large set of certificates an attacker may possibly leverage (with or without social engineering techniques). Also signed but revoked certificates can be used by an attacker because the clients are not designed to check a certificate revocation list.

4. ATTACK DESCRIPTION

Our attack uses a modified version of the software AP hostapd [20] to capture user login data and successfully authenticate users without possessing the users' authentication data.

In the following, we describe the general approach of the attack, the setup and necessary modifications to hostapd, and the technical characteristics of the used exploit.

4.1 Goals and Approach

The attack pursuits two main goals, which are to be accomplished subsequently:

1. Capture login data in order to gain network access and access to other services.
2. Authenticate the user on the malicious AP in order to perform a MITM-attack and gain control over his network traffic.

Capture Login Data. In order to achieve the first goal, the attacker configures a malicious AP to mimic a valid Eduroam access point by setting up the same SSID, creating a server certificate with an arbitrary CN string, and getting it signed by one of the valid CAs which have the same top-level authority as the RADIUS server certificate. The last one turns out to generally not cause problems because a common used certificate infrastructure for the Eduroam network is to have a large CA as a top level authority (for Germany the *Deutsche Telekom Root CA 2*). The top level authority commonly signs multiple intermediate CAs which often offer certificate signing services for server certificates. Client devices fulfilling the prerequisites from Section 3.1, especially prerequisite d), will connect to the attacker's AP and successfully finish the handshake phase because each communicating party in TLS is responsible for validating certificates [13].

The attacker thus exploits insufficient information checking, stemming from erroneous or misconfigured user clients. This enables the attacker to capture username and password for the inner PAP authentication or for user identity, authenticator challenge, and peer response when using inner MS-CHAPv2 authentication (see Section 2.2).

Authenticate Users. The second goal aims at devices that use wpa_supplicant [21] as supplicant software and have configured EAP-TTLS as authentication method (prerequisite a).

The attacker exploits the `eap_workaround` compatibility setting of wpa_supplicant to successfully authenticate the victim at his malicious AP. The attacker can now act as a MITM, providing network access by routing the victim over the malicious AP into the target network. For this purpose, the attacker can use either own authentication data if he has access to the target network or some other users' data captured in step 1 (capture login data).

4.2 Malicious AP Setup

For the attack a modified version of the open source software AP hostapd [20] is used. The hostapd version is executed on a Linux machine with a Xubuntu 13.10 AMD64 operating system equipped with two wireless NICs: One used by hostapd[1], and one for connecting with the target network. The wireless NIC used by hostapd is a TP-Link TL-WN7722N. Additionally an own DHCP server is set up to route victims into the target network.

As part of the attack, the following modifications are made to hostapd:

1. Modify user database access to accept every identity in order to capture login data.
2. Disable verification of authentication data in the invoked EAP methods of hostapd to authenticate the user on the malicious AP.

Additionally, hostapd is extended by a custom logger, which logs MAC-addresses and user credentials for cracking them later (a number of credential crackers exist, e. g., CloudCracker [22] or Asleap [31]).

Modifying database access. Database access modifications are required to capture user authentication data while using inner MS-CHAPv2, since the MS-CHAPv2 implementation in EAP-TTLS does not process challenge values when no match is found for the victim in the user database and aborts in PEAP after the identity request. Therefore it is necessary that hostapd accepts arbitrary user identities in the data phase.

During runtime, user database entries are stored in hostapd as elements inside a linked list. To accept arbitrary identities, a dummy user entry is introduced, serving as a placeholder for an arbitrary data phase identity. The function returning user entries from the list is modified by inserting a conditional break statement, so that it always returns the dummy user entry when a data phase user is queried.

Disabling verification. Disabling authentication data verification inside the EAP methods serves the purpose of successfully finishing user authentication in EAP-TTLS by making hostapd accept every submitted user password value in EAP-TTLS/PAP and EAP-TTLS/MS-CHAPv2.

In PAP, verification of authentication data is a simple comparison of submitted username and password to the values stored in database. Disabling the comparison to always return true enables the attacker to authenticate users in EAP-TTLS/PAP.

In MS-CHAPv2, hostapd has to accept the received NT-Response as valid by disabling the comparison with R_{Pcheck} in order to successfully authenticate unknown users (see Figure 5). Additionally the subsequent state of the inner authentication is changed to `success` to omit computation of the authenticator response, submit a success message, and thus exploit the `eap_workaround` setting. In the following section, we describe the technical characteristics and the impact of the exploit.

4.3 Exploit Characteristics

Handling of EAP is implemented as state machine (SM) similarly to [29] on hostapd and wpa_supplicant, with inner authentication running as a SM on top of each EAP SM.

[1]The wireless NIC must support AP-mode.

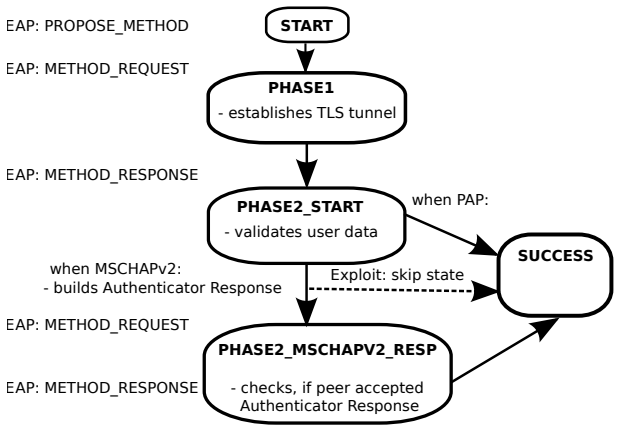

Figure 6: State flow for inner TTLS authentication on the server side. The dashed arrow indicates state skip in EAP-TTLS/MS-CHAPv2 used for the exploit.

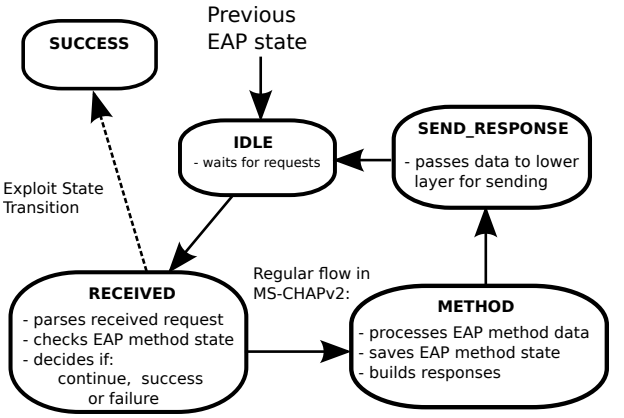

Figure 7: State flow for inner TTLS authentication with exploit on the client side. The dashed arrow indicates state transition triggered by a received EAP success after omitting the Authenticator Response in EAP-TTLS/MS-CHAPv2.

To authenticate Eduroam users in EAP-TTLS/PAP the modification described in Section 4.2 is sufficient due to the simple design of PAP (Section 2.2).

For EAP-TTLS/MS-CHAPv2, the `eap_workaround` compatibility setting of wpa_supplicant is exploited. This compatibility setting exists because of non-conforming authentication server implementations, one of them being the EAP-TTLS/MS-CHAPv2 implementation in a legacy version of FreeRADIUS [28].[2] As we could identify, this implementation omitted the concluding success message / authenticator response in EAP-TTLS (see Figure 5). In order to be operable with the legacy FreeRADIUS version, the default authentication state in inner MS-CHAPv2 of wpa_supplicant was set to conditional success after transmitting R_P. By accepting the provided R_P and additionally skipping the following state and directly advancing to success, the integrated authentication server in hostapd does not respond with the authenticator response and behaves like FreeRADIUS, transmitting an EAP success.

When wpa_supplicant receives the transmitted EAP success, the EAP-TTLS method returns to the calling EAP SM with conditional success and the authentication is successfully concluded by both parties without knowing the user password. The resulting state flows are depicted in Figure 6 for the server part and Figure 7 for the client. Figure 6 shows the EAP-TTLS server state machine for MS-CHAPv2, triggered by the EAP SM. After receiving the EAP Response and validating the contained user data, the exploit forces the EAP-TTLS SM to directly process to SUCCESS, instead of submitting the authenticator response and waiting in PHASE2_MSCHAPV2_RESP for the client to validate it. The server therefore passes the same state sequence as if using PAP and behaves as if the authentication was finished successfully by sending an EAP Success. The EAP Success message triggers the EAP SM of the client, depicted in Figure 7, into SUCCESS instead of proceeding to state METHOD and process the received data because it does not necessarily expect an authenticator response due to the `eap_workaround` setting.

The compatibility setting is enabled when the supplicant wpa_supplicant is run with default settings—as it is done in Android [1]—and must be deactivated by adding the option `eap_workaround=0` in the wpa_supplicant configuration file either by providing this option in the user interface or by having access to the configuration file. Unfortunately Wi-Fi configuration settings are highly restricted in Android such that this is not possible without having root access to the device.

Applicability. The described exploit was tested on eight different types of wireless devices. The results are presented in Table 1. As shown in the table, all tested Android-based devices prior to version 5.0 are vulnerable to the presented exploit. This is due to the infeasibility of deactivating `eap_workaround` in the Wi-Fi-settings without having root access to the device and usage of wpa_supplicant version 2.2 or older.

The exploit turned out to also work for Apple devices when the user can be tricked to accept the presented CA certificate and he has configured his device using the restricting user interface (UI) and not by pre-built configuration profiles.[3] When using the UI, the only parameters which can be set are username, password, followed by a CA certificate prompt.

Reporting and Bug Fixing. After the detection of the presented security issues, we followed the responsible disclosure policy and got in contact with the respective development teams. We can report the following results: *i*) The described problem was reported to the Android security team of Google on June, 12th 2014 and was fixed for the final release of Android 5.0 Lollipop. *ii*) The EAP-TTLS compatibility setting, which led to the presented vulnerability, was removed in wpa_supplicant version 2.3, released on October, 9th 2014.

[2]This compatibility setting is no longer necessary for recent versions, as tests with version 2.1.10 of FreeRADIUS, installed from Ubuntu 12.04 LTS packet archives, have shown.

[3]These profiles are created by Eduroam CAT [4] or Apple Configurator [2] and enable the user to set the CA certificate and configure otherwise unavailable settings, including authentication method, outer identity, and server name pinning (common name check, CN).

Table 1: Malicious AP Test results for various devices. Mac OS and iOS devices require acceptance of the server certificate by the user.

Device	Software version	EAP-TTLS/PAP	EAP-TTLS/MS-CHAPv2
HTC Desire	CM 7.2.0.1 (Android 2.3.7)	authenticates	authenticates
Tolino Shine	1.3.0 Rev.1722 (Android 2.3.7)	authenticates	authenticates
Samsung Galaxy S3	Android 4.3 TouchWiz	authenticates	authenticates
LG Google Nexus 5	Android 4.4.4	authenticates	authenticates
Sony Xperia mini pro	CM 10.2.0.1 (Android 4.0.2)	authenticates	authenticates
Ipad Mini (1st Gen)	iOS 7.1	N/A	may authenticate
Macbook Pro (2013)	Mac OS X 10.9.3	N/A	may authenticate
LG Google Nexus 4	Android 5.0.1	authenticates	does not authenticate (fixed)

Despite the fixes in Android 5.0.1 and wpa_supplicant 2.3, the issues presented in this paper remain highly relevant: At the time of writing, Android 5 has only a negligible market share according to the Google Platform Version statistics. In addition, the CN check configuration is still not accessible for the user on Android 5 devices[4]. Therefore Android 5 devices can still be authenticated in EAP-TTLS/PAP due to the simple protocol structure, although the exploit described in 4 does not work any more (see Table 1).

5. PRACTICAL STUDY

To find out how many devices in the field would be affected by insufficient device configurations, a practical study on an educational event in cooperation with the IT-Centre of the Ruhr-University Bochum was carried out. At this event, client device configurations were scanned remotely while providing information about a proper device configuration to members of the university and giving the chance to improve it. This section describes the setup of the practical study and its results.

5.1 Setup

For the practical study, a modified version of hostapd was used to remotely validate client device configuration, targeting members of the university. It was set up to mimic a valid Eduroam access point in a room that was largely isolated from other wireless networks to avoid interference with the Eduroam network and users not informed about the event. The group of participants consisted of volunteers who were invited to check their device configuration following a university-wide announcement. The entire event was conducted by the members of the university's IT-Centre. Affected users were displayed in anonymised form on a large screen and direct configuration help was offered at the event.

In technical terms, the used hostapd version was modified as follows:

- It filters the domain suffix for `@realm.tld` (domain of the university) or missing domain suffix and hence addresses only members of the university.
- It rejects every connection attempt and does not log key material and passwords.
- It detects usage of PAP as inner authentication method.
- It logs data in an SQLite database and distinguishes between correctly and wrongly configured devices.

[4]Filed as issue 74244 in the Android bugtracker.

Table 2: Results of the Practical Study

Category	Amount	Percent
Number of Users	$\sim$ 350	
Total devices	507	
Share Apple	193	38 %
Total wrongly configured	264	
Share using inner PAP	53	20 %
Share Apple	34	13 %

The logged data consists of timestamp, user identity, and MAC address, enabling clear identification of participants.

In order to distinguish correctly and wrongly configured clients, client classification based on stages reached in the authentication process was achieved by tracking the EAP SM state flow in hostapd (see Section 4.3). A device is vulnerable and considered wrongly configured once the TLS connection is established and the phase 2 identity is successful queried. When the anonymous identity was successfully queried and the peer rejected the presented server certificate, then the device is considered as configured correctly.

If the anonymous identity indicates an user from a foreign institution, then it is considered invalid and the corresponding client will be ignored and rejected before starting the TLS handshake.

To prevent multiple user entries due to connection reattempts after rejection, a check was introduced if an entry already existed in the SQLite database. If so, merely the timestamps were updated. The logged data was evaluated in anonymised form.

5.2 Results

The results of the practical study are presented in Table 2. They are based on data collected during the one-time educational event in a time period of approximately 3.5 hours, mid-day, during the semester. There was no selection of user devices, every provided device was accepted.

The event took place roughly five months after the instructions for the correct configuration of devices for using Eduroam were updated on the website of the IT-center and leaflets on this topic had been distributed to students and staff (regarding the use of MS-CHAPv2 instead of PAP and the correct installation of the root of trust for the certificate chain).

Since Apple devices turned out to be also affected in case EAP-TTLS/MS-CHAPv2 is used, their share on the total numbers is additionally listed in Table 2. They could be

identified based on the vendor part of the HW-address since Apple is one of few hardware manufacturers that uses an registered MAC address space for their devices.

The total amount of devices was determined by the total count of different MAC addresses. From a total of 507 devices, 52 % devices were vulnerable. A total 20 % of the vulnerable devices used PAP as inner authentication method and thus were leaking authentication data in unencrypted form.

6. DISCUSSION & COUNTERMEASURES

Our study showed that the majority of the users of the Eduroam network are vulnerable to our attack.

In the following, we discuss the results of our study and propose countermeasures to prevent the presented attack.

6.1 Discussion

As the results presented in Section 5.2 show, a share of 52 % wrongly configured devices existed in our study, months after the configuration manuals for Eduroam access had been updated. The comparatively small share of 13 % of wrongly configured Apple devices might be due to simplifications of the Wi-Fi configuration by importing pre-built configuration profiles.

Given the erroneous configuration manuals, it is also probable that the remaining 230 vulnerable devices were Android devices, although this cannot be determined with certainty, since the vendor part of the HW address does not enable to draw conclusions about the used OS in other cases than Apple. In the case that they were indeed Android-based, these devices could additionally fall victim to the attack described in Section 4, since `eap_workaround` deactivation in Android is not possible unless having root access to the device, which is very uncommon.

Note that the case study only examined wrongly configured client devices. It does not provide results about correctly configured devices that are vulnerable to this attack because they do not check the CN string of the certificate.

6.2 Countermeasures

There are multiple approaches to provide resilience against the attack described in this paper. We categorize the following countermeasures into *client-side* and *infrastructure-side*. Note that only two combinations of the proposed countermeasures (C1 and C2 or C1 and I1) are able to entirely prevent the attack, as shown in Table 3. The others only make the described attack more difficult.

Table 3: Effectiveness of Countermeasure Combinations. Effectiveness is divided into: "✔": Prevents attack entirely; "○": Only increases difficulty; "✕": Does not prevent attack.

	C1	C2	C3	I1	I2
C1	✕	✔	○	✔	○
C2	✔	✕	○	✕	○
C3	○	○	✕	✕	○
I1	✔	✕	✕	✕	○
I2	○	○	○	○	○

Client-side. Multiple ways exist to minimize the share of users with wrongly configured devices, thus reducing the attack surface. The following approaches provide resilience against the attack as long as the users are able or willing to make changes to their device configurations.

C1. **Correct client configuration:** According to the results presented in Section 5.2, 45% of the participants own (checked) a second wireless device in average. This number is likely to increase in the future, with every device being a possible security weakness because certificate validation is in responsibility of the client [13], configured by the user. This underlines the need of actively educating the user about the risks of using an unprotected connection and ways to configure their wireless devices properly, especially the CA certificate. Simplifying and streamlining the configuration process would make it less tedious in the users' eyes and increase their motivation to properly configure their devices. Automating the process, e. g., by using configuration profiles like on Apple devices, would help eliminate error sources. When automated features are not available, lowered restrictions in the UI should make already implemented supplicant features available for configuration by the user. Additionally, access points which automatically validate the used configuration of the client would help to eliminate the high number of incorrectly configured devices. We further elaborate on this idea in Section 6.3.

C2. **Activating CN check:** Activating the CN check helps to identify the attacker's rogue access point. Wpa_supplicant is capable of performing the CN check by setting the `subject_match` option in its configuration. However, this option is not reachable from the UI on Android devices and activation requires root access to the device. Therefore, it is only accessible to technically-minded users which have root access to their Android device.

On Apple devices the CN check is an option which is provided by the configuration profile and can be activated by installing the configuration profile for the wireless network. However, with a manual configuration of the wireless network access, the CN check is also not an available option to the user.

C3. **Deactivating `eap_workaround`:** In order to provide resilience against the exploit described in Section 4.3, the `eap_workaround` setting must be deactivated in the network entry of wpa_supplicant [21]. Disabling this compatibility setting makes wpa_supplicant expect the correct submission of the Authenticator Response R_A for validation in EAP-TTLS/MS-CHAPv2. This applies to Linux and Android, since wpa_supplicant is the default supplicant software in these operating systems, yet requires root device access in Android.

However, this countermeasure does not prevent the attacker from capturing the user credentials and using services like CloudCracker [22] to recover the used password.

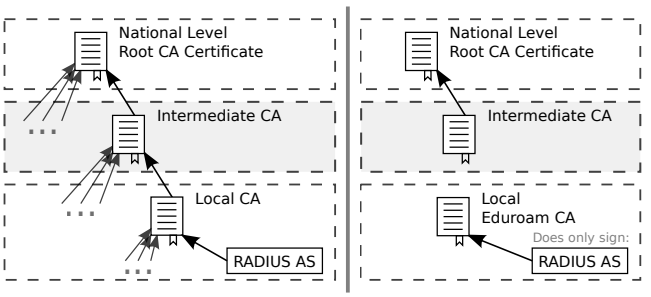

Figure 8: National certificate structure of Eduroam before and after applying countermeasure I1: On the left: Current situation with an indefinite amount of intermediate and server certificates signed by the same root CA and therefore valid for rogue authentication. On the right: Resulting structure after applying I1. Because the local Eduroam CA is only used for RADIUS AS, it is not possible anymore for an attacker to obtain a valid rogue certificate.

Infrastructure-side. Several approaches can be applied in order to prevent the attack by making changes on the used infrastructure. This is important due to the difficulty of enforcing or enabling the client-side countermeasures previously presented, leaving numerous client devices vulnerable against the attack described in this paper. Furthermore, we believe that a change on the server side of the infrastructure is simpler and more realistic than a software fix on each client, e. g., to enable the activation of the CN string check on all Android devices.

I1 **Changing the Eduroam certificate infrastructure:** Currently, the top level CA typically precedes a chain of signed intermediate CAs. One of the intermediate CAs signs the RADIUS server certificate.

Because the majority of clients are not able to check the CN string of the certificate offered by the RADIUS server (see also C2), even correctly configured clients are vulnerable to the attack described in this paper. The only requirement for this attack to be successful is that the certificate is signed by one of the intermediate CAs (or top level CA) in the certificate chain. The valid signed certificate can have an arbitrary CN string because it is not checked.

However, all clients are able to check the validity of the certificate with respect to the top level CA. Therefore, we suggest to change the used certificate infrastructure of the Eduroam network *to use a separated CA which is entirely used for the local Eduroam instance*, as depicted in Figure 8. With a separated Eduroam CA for each participating institution, which only signs the RADIUS server certificate, each correctly configured client is immune against this attacker. For instance, in the case of a university, the network operations center would generate its own Eduroam CA and sign the RADIUS server certificate exclusively. To work correctly, the Eduroam CA has to be installed and configured on each client that authenticates itself against the university's RADIUS server. Note that this process, i. e., the authentic distribution of certificates, must also be done in the current setting with the official top level CA. On most devices, i.e., Android devices, it is not possible to use the pre-installed well known CAs for the Wi-Fi settings. Hence, a used top level CA has to be manually installed in the same way a self generated Eduroam CA is. Though, a top level CA makes Eduroam vulnerable against the described attack.

During the authentication process, the client devices are routed to their home RADIUS instance and only establish a TLS tunnel with this server (Section 2.1). Because of this routing process, the proposed change to the Eduroam certificate infrastructure does not even affect the roaming capability. Therefore we consider it as the simplest and most effective way to prevent the presented attack.

I2 **Using Evil Twin detection techniques:** Since the attack setup introduces an additional AP which forwards traffic, the usage of rogue AP detection on the infrastructure and/or client side can help to forestall this attack. It would, however, introduce additional network overhead for detection mechanisms and require additional hardware. Countermeasures on the infrastructure side cover introducing probe traffic and sensors as described in [34] and [9], while monitoring network behavior and maintaining a host database. On the client side the usage of a hop-based rogue AP detection, like ETSniffer [27] or WiFiHop [24], could be used to prevent connecting to the rogue AP, introducing more responsibility and configuration effort to the user, which already is a problem regarding the certificate configuration. While these approaches state to have a very high success rate on detecting evil twin APs, they have been tested in corporate network scenarios but not in the Eduroam environment, which might have a more dynamic characteristic. Also this countermeasure would only detect an attacker rather than prevent the attack entirely.

6.3 Automated Configuration Validation

The Eduroam infrastructure has an exceptionally large user base and its security also relies on user cooperation, wherefore it is hard to prevent configuration errors.

In order to further lower the number of incorrectly configured client devices, we have modified the rogue software AP used in our attacks into a proof of concept configuration validator. The configuration validator is based on the software AP hostapd and mimics a valid Eduroam AP using a server certificate signed by an intentionally invalid "rogue" CA. Clients fulfilling the requirements and behavior presented in Section 3.1 will automatically connect to the AP which tracks the authentication process and derives the configuration status from the reached authentication phase:

- Phase 1: Rejecting the server certificate indicates a correctly configured client.
- Phase 2: Successful TLS tunnel establishment and user identity reception indicates wrong client configuration.

The AP logs the reached authentication phase and the used EAP method. In order to identify the user and his device, the hardware address of the device, the submitted identity, and the timestamp of the connection attempt are logged. However, the configuration validator never finishes authentication and does not log any key or password data.

Given this data, it is also possible to detect clients which do not use an anonymous phase 1 identity, e. g., Apple clients which were not configured using a configuration profile. Because the email address corresponding to the logged identity is known to the Eduroam service provider, the operator is able to notify the user about the deficient configuration of his device and can take further measures.

In order to make sure to monitor only local students, the AP filters the submitted identity in authentication phase 1 according to the realm suffix and immediately rejects connection attempts from visitors.

The AP runs on a headless Raspberry Pi Model B using an AP-enabled USB Wi-Fi NIC and can be integrated into the institution's network. To ensure data privacy, the device is encrypted and has to be securely unlocked over network on boot.

7. RELATED WORK

Online released versions of modified software AP and AS used for attacks and vulnerability detection are [32, 15, 7]. In [32] a rogue RADIUS authentication server was introduced which is also able to log credentials from EAP methods used in Eduroam. In comparison to the hostapd setup presented in this paper, it does not provide EAP-TTLS MSCHAPv2 user authentication and introduces an additional component by using the modified FreeRADIUS, which needs to be set up and running.

Hostapd was also used as diagnostic tool in [15] to detect vulnerability for the heartbleed exploit and for logging user data in [7], targeting EAP-FAST and PEAP/MS-CHAPv2 credentials.

Scientific publications related to our work include [8, 16, 26, 11, 10]. MITM attacks in tunneled EAP methods were discussed in [8] and [16]. In [16], EAP-MD5 and EAP-TTLS were subject to attacks using a modified hostapd with EAP-MD5 as targeted inner authentication. The MITM attack in [8] used a different setup: It captures data from clients using legacy authentication protocols, like MS-CHAPv2 or MD5, and tunnels the data into PEAP or EAP-TTLS sessions.

A MITM-attack setup that also authenticates users on a rogue AP without possessing user credentials is presented in [26]. It uses the LEAP authentication method as oracle for PEAP in order to authenticate the user on a rogue AP and targets especially Apple devices.

An elaborated attack against WPA2-Enterprise authentication was presented in [11], using UI shortcomings in common operating systems to mimic the target network and forged certificates and jamming techniques to disassociate the victim and thus making it connect to the fake network. In order to perform a MITM attack, the MS-CHAPv2 values of the inner authentication were cracked using a specific high performance system. This step is not needed in our case.

Work on certificate validation errors has been presented by Brubaker et al. [10]. They focused on the automated generation of fake certificates in order to fuzz the SSL/TLS implementations. Their results also show that established techniques still have erroneous implementations—an insight that we further substantiate in this paper.

8. CONCLUSION

This paper presented a twofold attack on the connection establishment and authentication process in Eduroam: capturing authentication data using an evil-twin access point and hooking up to the connection as a MITM, thus being able to capture and manipulate the victims' network traffic. The attack exploits the fact that many Eduroam users have a missing or incorrect CA certificate configuration on their wireless devices. Furthermore, the current national trust structure in Eduroam often enables an attacker to perform this attack despite correct device configurations. Tests with various mobile wireless devices have shown that every device whose Eduroam access is configured using default user-available configuration options is vulnerable to this exploit.

Our presented countermeasures show that a proper device configuration with an accessible CN check, and thus a correct certificate validation, is an effective countermeasure to prevent this attack. Due to technical restrictions and the vast amount of client devices, it is not possible to activate the CN check on a great share of devices, though. We therefore proposed an isolated local Eduroam CA as a simple change to the network's trust infrastructure. This CA does not affect the roaming functionality, yet turns out to be a highly effective countermeasure for the presented attack, since it is not possible for an attacker to obtain a valid server certificate for a rogue AP. Additionally, to detect and further minimize the amount of wrongly configured devices we have shown an automated way to check for such devices.

9. REFERENCES

[1] android Git repositories. online: `https://android.googlesource.com/`.

[2] Apple Business & Education Support. online: `https://www.apple.com/support/iphone/business/`.

[3] Devices that are compatible with eduroam. online: `https://confluence.terena.org/display/H2eduroam/How+to+deploy+eduroam+on-site+or+on+campus#Howtodeployeduroamon-siteoroncampus-Devicesthatarecompatiblewitheduroam`.

[4] eduroam Configuration Assistant Tool. online: `https://cat.eduroam.org/`.

[5] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz. Extensible Authentication Protocol (EAP). Technical report, IETF, June 2004.

[6] H. Andersson, S. Josefsson, G. Zorn, and B. Aboba. Protected Extensible Authentication Protocol (PEAP). Technical report, IETF, October 2001.

[7] B. Antoniewicz. Hacking EAP-FAST Phase 0 with hostapd-wpe. online: `http://blog.opensecurityresearch.com/2013/04/hacking-eap-fast-phase-0-with-hostapd.html`, 2013.

[8] N. Asokan, V. Niemi, and K. Nyberg. Man-in-the-middle in tunnelled authentication protocols. In *11th International Workshop of Security Protocols, Cambridge, UK, Revised Selected Papers*, pages 28–41, 2003.

[9] P. Bahl, R. Chandra, J. Padhye, L. Ravindranath, M. Singh, A. Wolman, and B. Zill. Enhancing the security of corporate wi-fi networks using dair. In *Proceedings of the 4th International Conference on Mobile Systems, Applications and Services*, MobiSys '06, pages 1–14, New York, NY, USA, 2006. ACM.

[10] C. Brubaker, S. Jana, B. Ray, S. Khurshid, and

V. Shmatikov. Using frankencerts for automated adversarial testing of certificate validation in SSL/TLS implementations. In *Security and Privacy (SP), 2014 IEEE Symposium on*, pages 114–129. IEEE, 2014.

[11] A. Cassola, W. K. Robertson, E. Kirda, and G. Noubir. A practical, targeted, and stealthy attack against WPA enterprise authentication. In *20th Annual Network and Distributed System Security Symposium (NDSS), San Diego, California, USA*, 2013.

[12] D. Dai Zovi and S. Macaulay. Attacking automatic wireless network selection. In *Information Assurance Workshop, 2005. IAW '05. Proceedings from the Sixth Annual IEEE SMC*, pages 365–372, June 2005.

[13] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. Technical report, IETF, August 2008.

[14] P. Funk and S. Blake-Wilson. Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0). Technical report, IETF, August 2008.

[15] L. Grangeia. Heartbleed, Cupid and Wireless. online: `http://www.sysvalue.com/heartbleed-cupid-wireless/`, 2014.

[16] H. Hwang, G. Jung, K. Sohn, and S. Park. A study on MITM (man in the middle) vulnerability in wireless network using 802.1x and eap. In *Information Science and Security, 2008. ICISS. International Conference on*, pages 164–170, Jan 2008.

[17] IEEE. IEEE Std 802.1X-2004. Technical report, IEEE, December 2004.

[18] V. Kamath, A. Palekar, and M. Wodrich. Microsoft's PEAP version 0 (Implementation in Windows XP SP1). Technical report, IETF, October 2002.

[19] B. Lloyd and W. Simpson. PPP Authentication Protocols. Technical report, IETF, October 1992.

[20] J. Malinen et al. hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator. online: `http://hostap.epitest.fi/hostapd/`, February 2014. [version 2.1].

[21] J. Malinen et al. Linux WPA/WPA2/IEEE 802.1X Supplicant. online: `http://hostap.epitest.fi/wpa_supplicant/`, 2014. [version 2.1].

[22] M. Marlinspike. Divide and Conquer: Cracking MS-CHAPv2. online: `https://www.cloudcracker.com/blog/2012/07/29/cracking-ms-chap-v2/`.

[23] M. Marlinspike. sslstrip. online: `http://www.thoughtcrime.org/software/sslstrip/`, 2009.

[24] D. Mónica and C. Ribeiro. Wifihop - mitigating the evil twin attack through multi-hop detection. In *Proceedings of the 16th European Symposium on Research in Computer Security (ESORICS), Leuven, Belgium, 2011*, pages 21–39, 2011.

[25] Mudge, B. Schneier, and D. Wagner. Cryptanalysis of Microsoft's PPTP Authentication Extensions (MS-CHAPv2). `https://www.schneier.com/paper-pptpv2.html`, September 1999.

[26] P. Robyns, B. Bonné, P. Quax, and W. Lamotte. Short Paper: Exploiting WPA2-enterprise Vendor Implementation Weaknesses Through Challenge Response Oracles. In *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless & Mobile Networks*, WiSec '14, pages 189–194, New York, NY, USA, 2014. ACM.

[27] Y. Song, C. Yang, and G. Gu. Who is peeping at your passwords at Starbucks? - to catch an evil twin access point. In *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Chicago, IL, USA*, 2010.

[28] The FreeRADIUS Server Project. The FreeRADIUS Project. online: `http://freeradius.org/`, 2014.

[29] J. Vollbrecht, P. Eronen, N. Petroni, and Y. Ohba. State Machines for Extensible Authentication Protocol (EAP) Peer and Authenticator. Technical report, IETF, August 2005.

[30] K. Wierenga, S. Winte, and T. Wolniewicz. The eduroam architecture for network roaming draft-wierenga-ietf-eduroam-01.txt. Technical report, IETF, February 2014.

[31] J. Wright. asleap Main. online: `http://wirelessdefence.org/Contents/AsleapMain.htm`.

[32] J. Wright. FreeRADIUS Wireless Pwnage Edition. online: `http://www.willhackforsushi.com/?page_id=37`, 2010.

[33] J. Yavor. The BYOD PEAP Show - Mobile Devices Bare Auth. online: `https://www.defcon.org/images/defcon-21/dc-21-presentations/Yavor/DEFCON-21-Yavor-The-BYOD-PEAP-Show-Updated.pdf`, August 2013.

[34] H. Yin, G. Chen, and J. Wang. Detecting protected layer-3 rogue aps. In *Broadband Communications, Networks and Systems, 2007. BROADNETS 2007. Fourth International Conference on*, pages 449–458, Sept 2007.

[35] G. Zorn. Microsoft PPP CHAP Extensions, Version 2. Technical report, IETF, January 2000.

[36] G. Zorn and S. Cobb. Microsoft PPP CHAP Extensions. Technical report, IETF, October 1998.