

UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Raul-Martin Rebane

Security of passwords in eduroam

Bachelor's Thesis (9 ECTS)

Supervisor: Dominique Unruh, PhD

Tartu 2016

Teenuse *eduroam* paroolide turvalisus

Lühikokkuvõte: Tartu Ülikool on otsustanud, et ülikooli traadita ühenduse kasutajanimi ja parool peab ühtima ülikooli kontos kasutusel oleva kasutajanime ja parooliga. See tähendab, et juhul kui ülikooli *eduroam* võrgul leidub mõni nõrkus, on seda potentsiaalselt võimalik ära kasutada kasutajate ülikooli kontole ligipääsuks. Antud uurimistöö on avastanud ühe sellise nõrkuse, milles luuakse võltsitud traadita ühenduse pääsupunkt, et saada kätte kasutaja autentimiseks kasutatava protokolliga kasutajapoolset vastust. Selle vastuse põhjal on ründajal võimalik kätte saada kasutaja parooli räsi, mida on omakorda võimalik kasutada Tartu Ülikooli Samba serveriga autentimiseks. Antud uurimistöö sisaldab *eduroami* ning rünnakus vaja minevate protokollide kirjeldusi ning ettepanekuid kuidas Tartu Ülikooli *eduroami* turvalisemaks muuta.

Võtmesõnad: eduroam, traadita võrguühenduste turvalisus, Extensible Authentication Protocol, MS-CHAPv2, Data Encryption Standard

CERCS: P170 - Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Security of eduroam passwords

Abstract: The University of Tartu has decided that the university's eduroam accounts will share the same user credentials as the rest of the university's services. This could potentially be abused by exploiting weaknesses in wireless security in order to gain access to a user's university account. The aim of this research was to uncover any such weaknesses. In the course of the research, an attack was discovered, which uses a spoofed access point to capture a handshake between the user and the authenticator, which can be used to retrieve a hash of the user's password. That hash is then used to authenticate to the university's Samba server. The thesis also provides the reader with details on how eduroam and the protocols used in the attack work, and discusses potential improvements to strengthen the security of Tartu University's eduroam.

Keywords: eduroam, wireless security, Extensible Authentication Protocol, MS-CHAPv2, Data Encryption Standard

CERCS: P170 - Computer science, numerical analysis, systems, control

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 4 |
| 2 | Background | 5 |
| 2.1 | Overview of eduroam | 5 |
| 2.2 | Extensible Authentication Protocol | 7 |
| 2.3 | MSCHAPv2 | 10 |
| 2.3.1 | Overview of MSCHAPv2 | 10 |
| 2.3.2 | DES and vulnerability in MSCHAPv2 | 12 |
| 2.4 | NTLM | 15 |
| 2.4.1 | Challenge response in NTLMv1 | 15 |
| 2.4.2 | Challenge response in NTLMv2 | 16 |
| 3 | Using Tartu University’s eduroam configuration to gain access to user files | 19 |
| 3.1 | Setting up a rogue access point | 19 |
| 3.1.1 | Windows 10 | 21 |
| 3.1.2 | Linux | 23 |
| 3.1.3 | Android | 23 |
| 3.2 | DES key cracking | 23 |
| 3.3 | Authenticating to Samba | 24 |
| 3.4 | Potential improvements for eduroam and UT | 25 |
| 4 | Results | 27 |
| 5 | Teenuse <i>eduroam</i> paroolide turvalisus | 28 |
| | Appendices | 34 |
| 1 | NetworkManager eduroam configuration | 34 |
| 2 | Descracker output | 35 |
| 3 | math.ut.ee Samba shares | 37 |

1 Introduction

Tartu University has decided that in order to make things more convenient for users their eduroam usernames and passwords will match those of other UT services such as ÕIS. However, convenience often comes at the expense of security. This research was conducted in order to determine whether this merge of passwords has an impact on security, with the aim to identify any potential attacks. During research different attack vectors were considered and tested. An attack was discovered which allows an attacker to gain access to the University of Tartu's Samba servers by exploiting the authentication protocols used in eduroam and UT's Samba server. This attack allows the attacker to gain access to any files shared with the user whose credentials were captured, including their UT home folder. Additionally, the attacker gains the user's NT Hash, which can then be used to potentially uncover the user's password. Details about unsuccessful attack vectors are not included in this thesis.

The first section contains details about eduroam as well as the protocols used in the attack. The eduroam subsection describes requirements set on its member universities and the infrastructure of the confederation itself. The following subsections outline the Extensible Authentication Protocol, an authentication framework required by eduroam; PEAP, the authentication protocol used in the UT eduroam configuration; MS-CHAPv2, the inner authentication protocol used in PEAP which is the root of the discovered attack; and NTLM, an authentication protocol currently in use in University of Tartu's Samba servers.

The second section contains details about the uncovered attack, beginning with a brief overview. The first subsection describes the process of setting up a rogue access point, which is used to capture the authentication handshake of the user. This subsection also details how different operating systems behave in different scenarios in the presence of this rogue access point. The second subsections describes the difficulty of brute forcing the aforementioned block cipher, a necessary process in this attack. The third subsection briefly explains Samba and demonstrates how to authenticate to University of Tartu's Samba using the password hash gained from previous subsections. The fourth and final subsection outlines different methods which can be used to improve the security of eduroam against this attack.

The third section contains the results of the thesis and their affect on Tartu University.

2 Background

2.1 Overview of eduroam

Eduroam, short for EDUcation ROAMing, is a service which allows users from participating academic institutions to gain secure internet access while visiting another participating academic institution [MM12]. The eduroam Policy Service Definition describes the goal eduroam user experience as "open your laptop and be online". Eduroam is based on the concept that in order to gain internet access in the university that the user is visiting, the authentication to the local network is carried out in the user's home institution using the home institution's specified authentication methods.

Eduroam is based on a hierarchical tree of Remote Authentication Dial-In User Service (RADIUS) servers, which manage the authentication of users. The tree's hierarchy allows the visiting user's authentication request to be routed to its home institution's RADIUS server. Users are given usernames in the form of `name@realm` where "realm" usually corresponds to the DNS domain name of the user's home institute. The institute's domain name is used to route the request within the tree. The leafs of the tree are the institutions' RADIUS servers. Those servers are connected to a national RADIUS server which in turn is connected to a European or global RADIUS server. As eduroam uses the country code portion of the domain name for routing, generic top-level-domain realms (such as in the case of `user@terenas.org`), where the user's country is impossible to determine, are handled as special cases and their use must be specifically requested. Figure 1 displays the different layers of the hierarchy with four institutes in two countries.

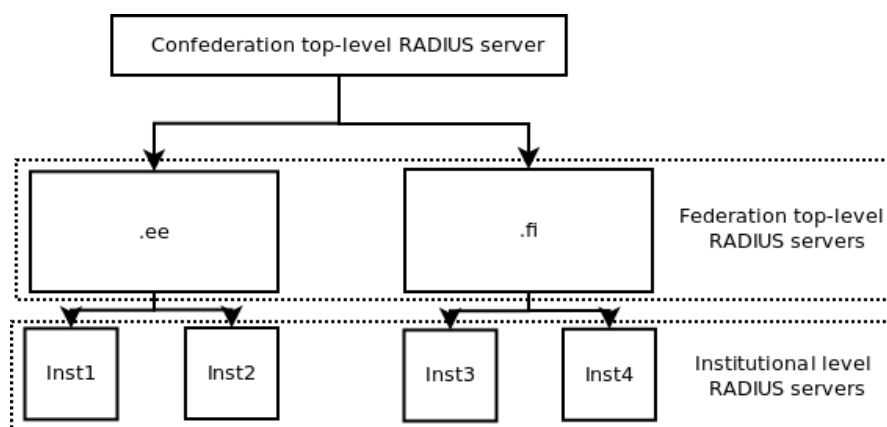


Figure 1: Layers of the eduroam RADIUS hierarchy [gro08].

The following describes the process where a user, who is visiting another insti-

tute within the confederation, authenticates to the local network via eduroam [Net].

- The user's device signals the authenticator, usually a wireless access point, that it wishes to access the internet.
- The authenticator signals the user that it wishes to initiate an Extensible Authentication Protocol (EAP) conversation.
- The user's response is forwarded by the authenticator to the local RADIUS server, where it is additionally forwarded along the eduroam network of RADIUS servers until it reaches the user's home university's RADIUS server.
- An encrypted connection is established between the user and the user's home university's RADIUS server.
- An EAP conversation is carried out.
- Once authentication via EAP is successful, the authenticator is notified. It then starts a handshake with the user to initiate the flow of encrypted wireless traffic.

Eduroam currently requires the use of EAP-TLS, EAP-TTLS or PEAP as end-to-end encryption support is required from each participating institution. This requirement is put in place to ensure that eduroam infrastructure operators are unable to steal the digital identities of users. In addition to requiring end-to-end encryption, eduroam also requires the participating institutions to provide technical data about the service, provide a user-support service to end users, and maintain a website which features a guide in English on how to use eduroam. While there are some technical specifications which the eduroam confederation requires institutions to follow, there are no other restrictions set in terms of authentication methods other than requiring the use of EAP-TLS, EAP-TTLS or PEAP.

2.2 Extensible Authentication Protocol

EAP

EAP, short for Extensible Authentication Protocol, is an authentication framework described in RFC-3748, which specifies different message formats to be used in protocols [Lev04]. Unlike previous Point-to-point protocols, where an authentication mechanism is chosen during the initial link establishment phase, in EAP an authentication mechanism is chosen in the authentication phase [Mic08]. These authentication methods are known as EAP methods. Once a method has been chosen, the peer and the authenticator begin an EAP conversation, which varies in length and format depending on the chosen EAP method. By the end of the conversation, the authenticator must transmit either an EAP Success (Code 3) message or an EAP Failure (Code 4) message.

The extensibility of EAP stems from its methods. To add a new EAP authentication method, both the authenticator and the peer only need to install an EAP method library file. The Internet Assigned Numbers Authority has specified over 40 EAP methods while new methods, such as EAP-Kerberos, are currently being drafted [Aut15, Rei16].

As EAP is a framework, not a protocol, there are several standards which define the encapsulation of EAP messages on different kinds of media. The IEEE 802.1x standard defines the encapsulation of EAP over LAN networks and the 802.11i standard defines EAP for wireless LAN's [Lev04, Abo05]. Use of EAP in conjunction with a pre-shared key is also referred to as WPA2-Enterprise [Sys08].

PEAP

EAP was designed for wired networks, which assumed physical security [Jos03]. As a result of this, EAP has no measures with which to protect the EAP conversation from attackers in cases where access to the medium is possible, such as over wireless media or IP. Furthermore, EAP Success and Failure packets are not authenticated, which means they can be forged by an attacker. Forged EAP Failure packets can then be used to force EAP peers to disconnect from the network. Forged EAP Success packets can be used by an attacker to convince a peer to connect to its network. To solve this issue, Cisco, Microsoft, and RSA Security developed PEAP, a protocol which encapsulates EAP within a TLS tunnel, which protects the EAP conversation by encrypting it and providing defense against spoofed Success/Failure packets by transmitting them within TLS.

In the following description, the term "peer" refers to a network access client and the term "server" refers to the authentication server, which is typically a RADIUS server [Mic15]. The network access point only relays messages between

the server and the peer and, if the server sends the appropriate message to the access point, initiates the handshake with the user to begin encrypted wireless traffic.

PEAP consists of two parts. The first stage establishes the TLS tunnel between the peer and the server. PEAP requires the server to have a server certificate to establish this connection. Client-side certificates are not required, which makes PEAP more convenient to use. Once a key has been negotiated, it is used to encrypt the rest of the conversation between the authenticator and the peer. Within the tunnel begins stage two, where a new EAP conversation is started for authentication, unless the peer had used a client-side certificate in the first stage. This process is illustrated by figure 2. This EAP conversation is carried out until the user is authenticated. EAP Extensions TLV - Success messages are exchanged within the tunnel to securely verify the success of the authentication, cleartext EAP Success messages can be tampered with.

While PEAP supports many inner authentication methods, MSCHAPv2 is by far the most common one [CB12]. The inner authentication method is chosen by a negotiation process. The server offers the peer an authentication method which the peer can either accept or refuse. If the peer accepts a method which the server suggested and supports, it will be used [Sys]. Otherwise, the authentication attempt will be cancelled by the server.

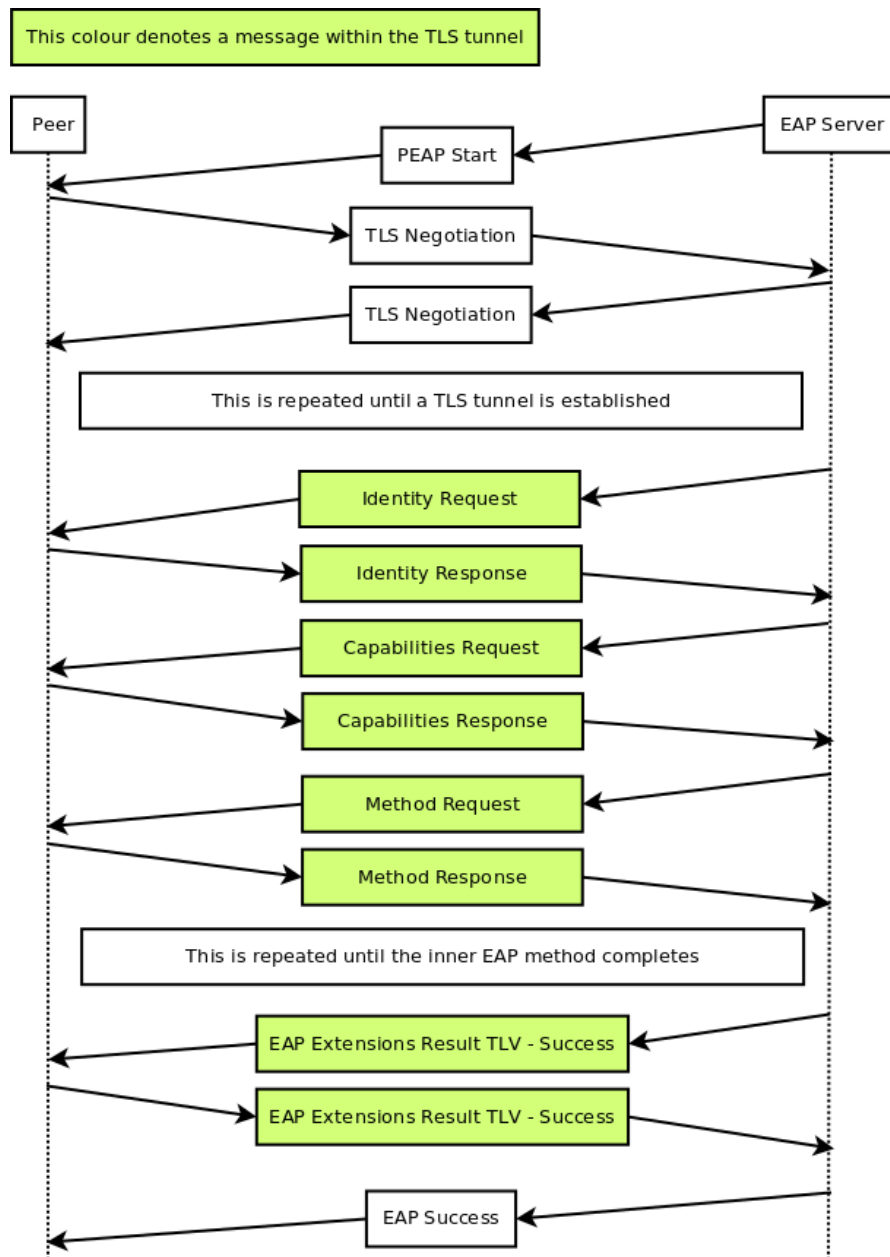


Figure 2: An example of a successful PEAP authentication [Mic15].

2.3 MSCHAPv2

2.3.1 Overview of MSCHAPv2

The Challenge-Handshake Authentication Protocol (CHAP) was created as a way to verify the identity of users by using a three-way handshake [Sim96]. The handshake would require both parties to generate responses which are based upon a secret, which has been previously shared and stored in plain. The advantage of this method is that neither party broadcasts the secret during authentication. The general steps of the Challenge-Handshake Authentication Protocol are as follows [Sim96].

- The authenticator sends a unique and unpredictable "challenge" message to the peer.
- The peer generates a value based on the shared secret and the received "challenge" message. It then sends it to the authenticator.
- The authenticator generates a value based on the shared secret and the "challenge" message. It then compares the value it generated to the one it received from the peer. If the values match, then the authentication is acknowledged and the authenticator sends a "success" packet.

In 1998, Microsoft released their own version of the CHAP protocol, named MS-CHAP, which allowed the authenticator to use hashes of the user's password instead of storing plaintext passwords [Cob98]. In 2000, an improved version was released and named MS-CHAPv2 which also provided mutual authentication, meaning that the authenticator's knowledge of the secret was required for a successful handshake. This was achieved by adding a "challenge" message sent by the peer alongside its "Response" packet and bundling the authenticator's response along with its "Success" packet [Zor00]. MS-CHAPv2 is still used to this day as the most common PEAP authentication method [CB12].

The MSCHAPv2 protocol works as follows [Zor00]:

- The authenticator generates a random 16-byte "Authenticator-Challenge" and transmits it to the peer.
- The peer generates a 16-byte "Peer-Challenge".
- The peer generates a "ChallengeHash" by hashing the concatenation of the "Peer-Challenge", the "Authenticator-Challenge", and the username. The hashing is done using the Secure Hash Algorithm 1 (SHA1).

- The peer generates a 16-byte NTHash of the user's password, which is the MD4 hash of the user's password in unicode.
- The peer pads the NTHash with zeroes to 21 bytes. The peer then splits this padded 21-byte block into three 7-byte blocks which will be used as three DES keys. The peer creates a "ChallengeResponse" by encrypting the "ChallengeHash" using the DES algorithm with the three keys and concatenating the results.
- The peer then sends the authenticator the "ChallengeResponse", the "ChallengeHash", the "Peer-Challenge", and the username. Figure 3 illustrates the steps described up to this point.

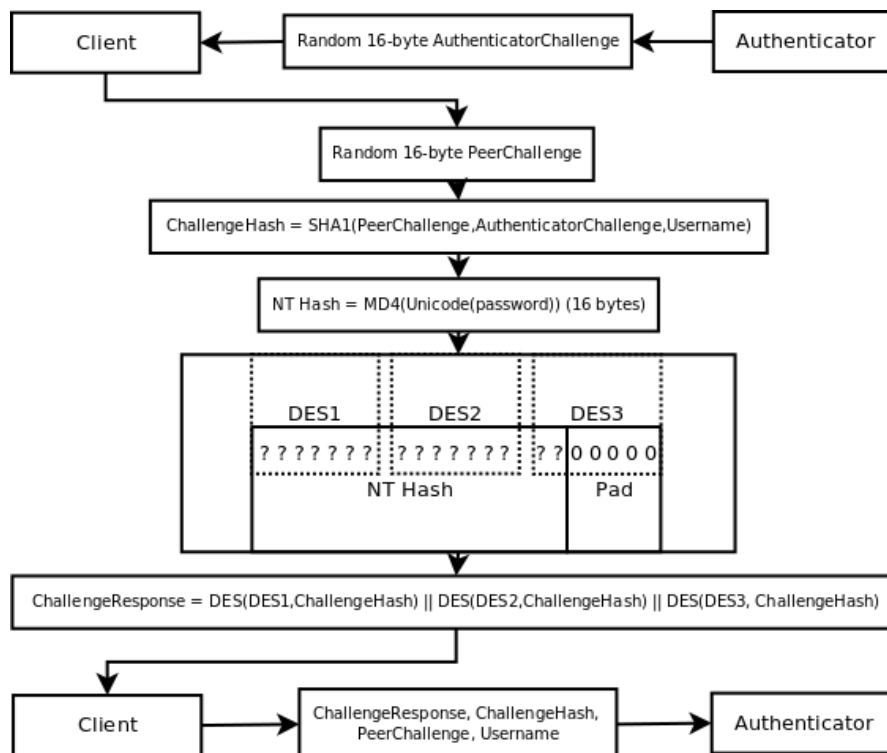


Figure 3: The generation of the ChallengeResponse [Mar12].

- The authenticator checks "ChallengeResponse" against the expected result. If they match, the authenticator will begin to generate the response that is required for mutual authentication.
- The authenticator hashes the NTHash of the user's password once more, creating a "PasswordHashHash".

- The authenticator hashes the "PasswordHashHash", the "ChallengeResponse", and a certain constant to create a "Digest". This hashing is done using SHA1
- The authenticator generates another hash called "AuthenticatorResponse" with the SHA1 function, using a concatenation of the "Digest", the "PeerChallenge", and another constant. The generation of the "AuthenticatorResponse" is illustrated by figure 4.
- 20 bytes of the "AuthenticatorResponse" are sent to the peer in the form of "S=", followed by 40 ASCII hexadecimal digits of "AuthenticatorResponse".

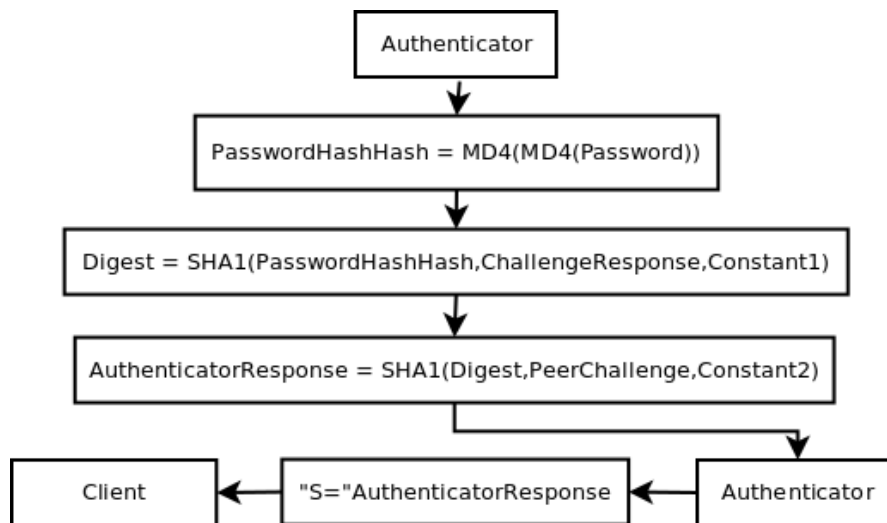


Figure 4: The generation of the AuthenticatorResponse [Mar12].

- The client then verifies the identity of the authenticator by replicating the above steps and comparing the result to the "AuthenticatorResponse".

2.3.2 DES and vulnerability in MSCHAPv2

The Data Encryption Standard (DES) algorithm used in MS-CHAPv2 is a block cipher which uses a single 7-byte key to both encrypt and decrypt 8-byte blocks of data into 8 bytes of ciphertext [Bur]. It was developed by IBM with the aid of the NSA and published in January of 1977 as a Federal Information Processing Standard for the United States [Int, Joh98, oST].

The DES became the most widely used encryption algorithm, both in commercial use and as a target of a generation of cryptanalysts. The DES proved

to be resilient against cryptanalytic attacks, but is now considered insecure since advancements in technology have made a brute force attack against a 7-byte key possible [Bur]. On June 18th 1997, the DESCHALL project successfully cracked a DES-encrypted message with the help of thousands of Internet users who performed a coordinated search of the key space which lasted three months [MC98]. With the use of specialized hardware, the time required for a brute-force attack was reduced to mere days in 1998 [Fou98a, Fou98b]. This specialized hardware consisted of a normal personal computer, which was connected to about 1500 custom-made "Deep-Crack" chips. These chips were designed to dismiss a large amount of incorrect keys in parallel, reducing the amount of potential keys to a fraction of the original key space. Software in the personal computer coordinated the chips and checked the small amount of undiscarded keys for the right one.

In July 2012, a man going by the pseudonym Moxie Marlinspike published an article for cracking MS-CHAPv2 by using a single search of the DES key space [Mar12]. The reason this is possible is that everything that the peer needs for authentication is transmitted, with the exception of the NTHash of the user's password. Figure 5 illustrates this by coloring the values which are not sent as red.

Since the unknown NT hash was only 16 bytes, it had to be padded with zeroes to 21 bytes. Then it was used as the basis for three DES keys, which encrypted the "ChallengeHash" into three ciphertexts. Those ciphertexts were then concatenated to form the "ChallengeResponse". Since both the "ChallengeResponse" and the "ChallengeHash" are sent from the peer to the authenticator, it is possible to find the three DES keys which were used to transform the "ChallengeHash" to parts of the "ChallengeResponse". By concatenating the keys, the attacker would then gain the NT hash of the user. As the third key only contains two significant bytes with the rest padded with zeroes, it takes very little time to find the right key. Finding the first and second keys is possible by iterating over the DES key space and encrypting the ChallengeHash with each key. The resulting ciphertext is then compared to the appropriate sections of the "ChallengeResponse", meaning that both keys will be retrieved without any additional DES operations compared to cracking just one key [Mar12]. With the increase in computing power since the late 90's, cracking DES has become much more affordable - Moxie Marlinspike provides a service named "Cloudcracker", which claims to perform this very attack within 21 hours for the price of 100 dollars [Int16]. This claim is, however, unverified. As of May 11th 2016, Cloudcracker is offline whereas it had been online in March 2016. It is unclear if the service is experiencing temporary difficulties or if the project has been cancelled.

As the peer is authenticated before the authenticator, anyone who attempts to connect to a rogue authenticator using MS-CHAPv2 is therefore providing the

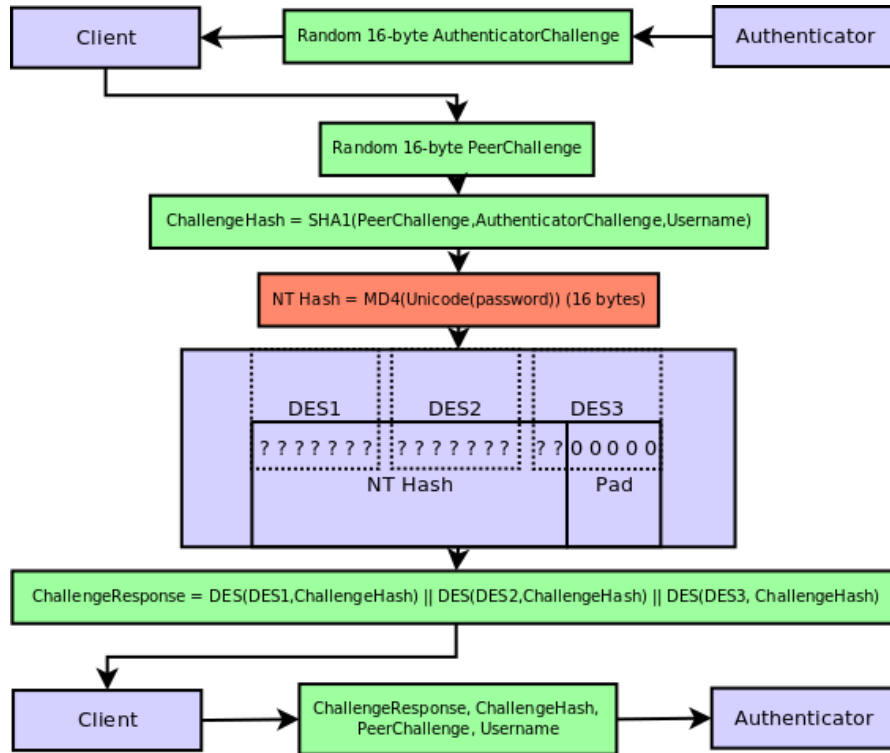


Figure 5: The unknowns of generating the ChallengeResponse [Mar12].

attacker with everything needed to recover their NTHash. Once the NTHash is retrieved, the attacker can then

- use it to authenticate to the MS-CHAPv2 network as the user;
- use it to perform the second authentication of MS-CHAPv2, appearing to the user as a legitimate network, and thus perform a man-in-the-middle attack;
- use the NTHash in other services which require it for authentication.

2.4 NTLM

Note: Reading this section is only required to fully understand why access to the NT hash allows authentication to the UT Samba service.

NTLM, short for NT Lan Manager, is a challenge-response authentication protocol used in Windows [Cor15]. Since the Windows 2000 Server operating system, the Kerberos authentication protocol replaces NTLM as the preferred authentication method. Despite that, NTLM support is still maintained for instances in which Kerberos is not suitable, for example if the server is not joined to a domain. As NTLM does not support recent cryptographic methods, Microsoft has advised applications to not use NTLM [HK97].

NTLM has several different versions - LM (Lan Manager), NTLMv1, NTLMv2 and NTLMv2 session. NTLMv2 session provides the option of extended session security, which provides message integrity and confidentiality. However, it is a misnomer as the protocol itself is NTLMv1 using the session security features which are also present in plain NTLMv2. To allow hosts to determine a minimum level of security, 6 different authentication levels have been specified. As the LM protocol does not use the user's NT hash to generate the challenge response packet, the specifics of generating LM response with a one way function will not be described.

2.4.1 Challenge response in NTLMv1

In NTLM, both an NT challenge response and an LM challenge response is sent. Depending on the security level, the LM response may not be required. In the NTLMv1 protocol, these responses for a non-anonymous client are generated as follows:

- Similarly to MSCHAPv2, the NT hash is padded with five zeroes and then split into three DES keys.
- If extended session security is enabled, the plaintext is set to the first 8 bytes of the MD5 hash of the concatenation of the 8-byte ServerChallenge and a random 8-byte ClientChallenge. This is then encrypted with the three DES keys with the results concatenated, resulting in a 24-byte NT challenge response. The LM challenge response is just set to a concatenation of the 8-byte client challenge and a 16-byte pad of zeroes.
- If extended session security is not enabled, the plaintext is simply the 8-byte ServerChallenge. It is then encrypted with the three DES keys and the concatenation of the ciphertexts becomes the 24-byte NT response. The LM challenge response is either calculated or set to the NT response value, depending on whether or not the value was required.

Figure 6 visualizes the described process. Note that if the LM challenge response is not required, the NT hash of the user is sufficient to generate the LM and NT responses.

2.4.2 Challenge response in NTLMv2

In NTLMv2, the NT and LM challenge responses are generated as follows:

- The 1-byte identifier of the response version, the 1-byte identifier of the highest possible response version, a 6-byte pad, the 8-byte little-endian GMT time, the 8-byte random client challenge, a 4-byte pad, the server name received from a previous packet, and another 4-byte pad are all concatenated to form a value labelled "temp".
- A HMAC-MD5 algorithm is used on the concatenation of the unicode uppercase username and the user's domain. The key for this operation is the NT hash of the user's password. This allows the server to check the integrity of the hash with the use of a shared secret key [HK97]. This value is named the ResponseKey.
- The server challenge and "temp" are then concatenated and then hashed using the HMAC-MD5 algorithm, with ResponseKey as the key. The NT response is the result of concatenating "temp" to the end of the 16-byte hash.
- The 8-byte server challenge and 8-byte client challenge are concatenated and hashed with the HMAC-MD5 algorithm, with ResponseKey as the key. The client challenge is concatenated to the end of the resulting hash. The result is the LM response.

Figure 7 visualizes the described process. Note that the NT hash of the user is sufficient to generate the LM and NT responses.



Figure 6: Generating the NT and LM responses in NTLMv1 [Cor15].

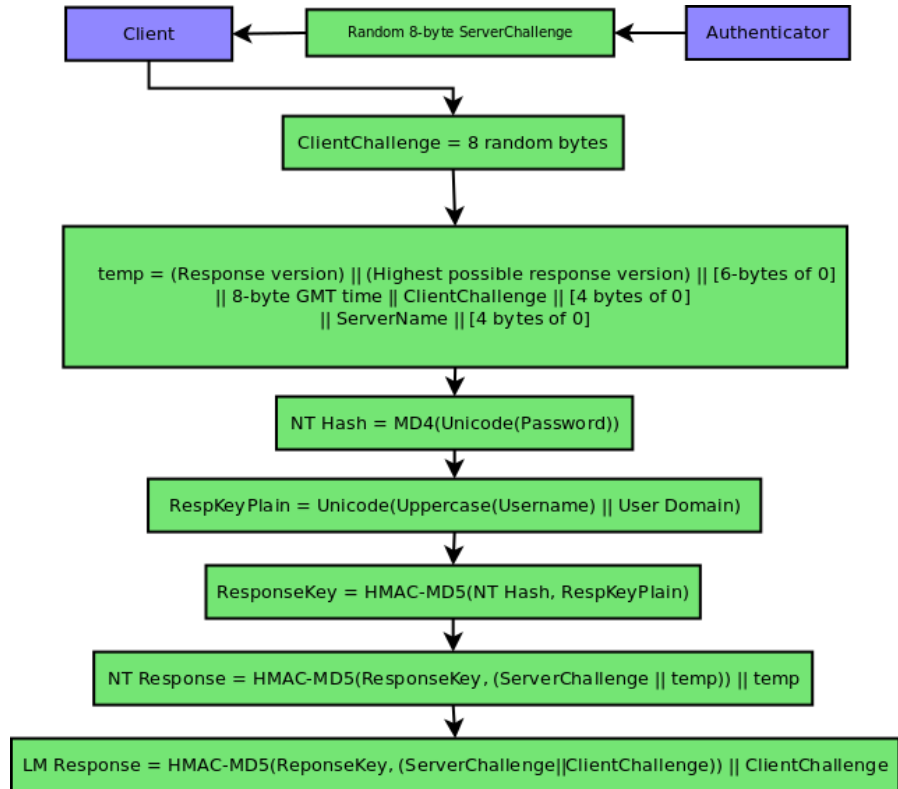


Figure 7: Generating the NT and LM responses in NTLMv2 [Cor15].

3 Using Tartu University’s eduroam configuration to gain access to user files

Tartu University’s eduroam uses a PEAP-MSCHAPv2 configuration. Multiple guides provided by the university on how to connect to eduroam also currently instruct students and faculty to configure their devices so that the PEAP server-side certificates are not validated [Kal16a, Kal16b, RK16]. When asked to comment, Erkki Kukk, a University of Tartu employee who was originally tasked with setting up eduroam in UT 10 years ago, stated the following: "We had to find balance between security and usability. In windows 2000, XP and 7 and maybe some other operating systems too you first had to have internet connection in order to download and install radius server certificate to the client machine. Since its world-wide project we also followed other institutions guides. I think for those reasons we decided to uncheck the certificate verification." However, this creates a potential weakness, as clients who do not validate certificates and attempt to connect to a spoofed access point will provide an attacker with everything required to retrieve the NT hash of the user’s password. This section describes an attack in which:

- the attacker creates a spoofed access point in order to capture the authentication reponse of the user;
- the attacker retrieves the NT hash of the user based on the captured authentication response;
- the attacker authenticates to the university’s Samba server using the retrieved NT hash.

The section also contains a subsection which contains suggestions on how to improve the security of Tartu University’s eduroam configuration and reduce the threat of this attack.

3.1 Setting up a rogue access point

If server certificate validation is disabled, identifying networks becomes a lot more difficult, as there are very few variables that the client can use to distinguish networks from one another. In NetworkManager, a network configuration tool used in several widely used linux distributions such as Debian, Ubuntu and Fedora, the only things that distinguish unverified networks from one another are the network’s ID and its authentication method [The16]. An example of the details saved by NetworkManager can be seen in appendix A. To successfully impersonate a

legitimate UT eduroam access point, the attacker's network needs to use PEAP-MSCHAPv2 and use the id "eduroam". To use EAP authentication, the attacker must set up a RADIUS server with which the client will communicate. As the MSCHAPv2 exchange in PEAP is encrypted in a TLS tunnel, extracting the required information would prove difficult without the help of specialized software. FreeRADIUS-WPE, a patch of the open-source RADIUS server software FreeRADIUS, provides a solution for this issue [Ant, Pro]. In the following examples, FreeRADIUS-WPE 2.1.12 is used.

If FreeRADIUS-WPE is installed, any authentication attempts will be visible in /usr/local/var/log/radius/freeradius-server-wpe.log. For example, a client with the username "test" and password "test" attempting to authenticate using PEAP-MSCHAPv2 will appear in the logs as follows:

```
mschap: Sun May  8 22:01:56 2016

username: test
challenge: d5:07:12:00:82:5e:df:44
response: 04:64:89:97:29:6b:5c:eb:86:d3:e1:65:fd:09:8d:b1:92:1c
          :59:21:6d:d0:ed:ab
john NETNTLM: test:
           $NETNTLM$d5071200825edf44$04648997296b5ceb86d3e165fd098db1921
           c59216dd0edab
```

Tests were ran to see how different operating systems behaved in the presence of rogue access points. This is designed to simulate an attack in which the attacker brings a rogue access point into the legitimate university network. In order to perform this in a controlled environment, two RADIUS servers were set up with two access points - one to signify the university's eduroam network and one to signify the attacker's access point.

A wireless access point was set up and connected to a RADIUS server running FreeRADIUS 2.2.8 on Ubuntu 16.04. This access point is henceforth referred to as RAD-LEG and it signifies the UT eduroam network. A wireless access point was set up and connected to a RADIUS server running FreeRADIUS-WPE 2.1.12 on debian 8 (jessie). This access point is henceforth referred to as RAD-ROG and it signifies the attacker's access point. Both access points were set to broadcast a network called "eduroam-test" with the PEAP-MSCHAPv2 authentication protocol.

A test account was set up by appending the following to /etc/freeradius/users:

```
testuser  Cleartext  Password:="eduroampassword"
```

To configure PEAP-MSCHAPv2 on RAD-LEG, "default_eap_type = peap" was set in the eap section of /etc/freeradius/eap.conf. In order for the access point to access the RADIUS server, it needs to be configured in /etc/freeradius/clients.conf.

To do so, the IP address and a secret between the access point and the RADIUS server were specified as follows:

```
client <AP IP> {  
    ipaddr = <AP IP>  
    secret = <secret phrase>  
    nastype = other  
}
```

Similar configuration was performed on RAD-ROG. Note that the RADIUS root folder on RAD-ROG is `/usr/local/etc/raddb` instead of `/etc/freeradius` as it uses a different version of FreeRADIUS.

3.1.1 Windows 10

By default, Windows 10 attempts to validate server-side certificates. The guides on the UT wiki site actively disable this feature. The following section describes different authentication use cases under the Windows 10 operating system.

In cases where

- the user has never connected to eduroam;
- the user has not configured the network using the UT guide;
- both RAD-ROG and RAD-LEG are in range;

only the access point with a stronger signal is displayed. If the user connects by double-clicking on the network, there will be an unverified certificate warning regardless of which access point is closest. Figure 8 illustrates the certificate warning displayed in Windows 10. If the strongest access point happens to be RAD-ROG, it is possible for the rogue access point to request that the inner authentication method would be PAP instead of MSCHAPv2, an insecure protocol which would give the attacker access to the user's password in plaintext.

In cases where

- the user has connected to eduroam previously;
- the user did not follow the UT guide and simply selected the legitimate eduroam access point correctly on their first connection;
- both RAD-ROD and RAD-LEG are in range;

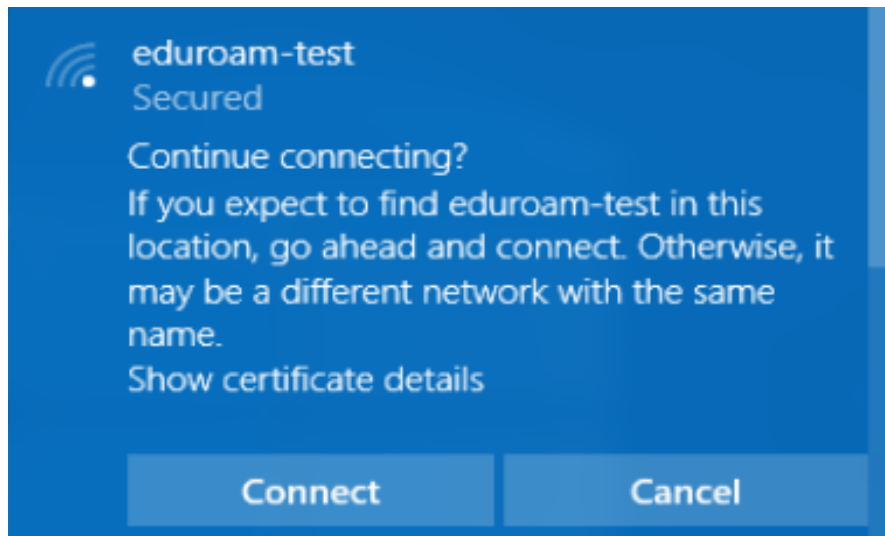


Figure 8: A Windows 10 certificate warning.

only the access point with a stronger signal is displayed. If the strongest access point happens to be RAD-LEG, everything will proceed without interruption and the user will connect to the legitimate network. If the strongest access point happens to be RAD-ROG, a certificate warning will be displayed. If the user dismisses the warning, the device will authenticate with RAD-ROG and the MSCHAPv2 handshake is retrieved. Note that the user is unable to access RAD-LEG and RAD-ROG is their only option, meaning the attacker either gains the MSCHAPv2 handshake or successfully performs a denial of service attack.

In cases where

- the user is connected to RAD-LEG;
- the user enters into an area where RAD-ROG has stronger signal than RAD-LEG;
- the user's device loses signal to RAD-LEG, causing reconnect attempts;

the device will attempt to connect to RAD-ROG. However, as RAD-ROG's certificate does not match, the PEAP-MSCHAPv2 authentication ends before the inner MSCHAPv2 stage. The user is left without a connection as they cannot reconnect to RAD-LEG.

In cases where

- the user has configured their network using the UT guide;
- both RAD-LEG and RAD-ROG are in range.

In these scenarios it does not matter whether the user is connecting for the first time or not. Whenever there is a connection being established, the access point with the strongest signal is chosen. There are never any certificate warnings displayed.

Note that even without connecting to a third-party Certificate Authority, the certificates are stored in the user's computer and thus the device is able to tell whether or not the user is trying to connect to a different network than normal and displays a warning accordingly. Following the UT guides in this case is less secure. However, this warning is most likely easily dismissed by users. For details, see "Validating the server certificate" paragraph on page 25.

3.1.2 Linux

Tests similar to the Windows 10 tests were ran using the Lubuntu 15.10 operating system with NetworkManager 1.0.4, abbreviated as NM. The key difference between Windows 10 and NM is that NM by default does not attempt to validate certificates. Therefore in cases where the user does not manually configure the certificate (meaning both default and UT-specified configuration), the device will always attempt to authenticate to whichever access point is nearest. In cases where the user does manually configure the server certificate, NM will not display a warning when the certificates do not match. The authentication attempt will then be stopped before the inner authentication method begins. Also, if the user is connecting for the first time, a prompt will be displayed which allows the user to specify the inner authentication method used in PEAP, with MSCHAPv2 being the default. This avoids the problem Windows has with the possibility of PAP authentication for first-time unconfigured users.

3.1.3 Android

Tests were ran using Android 5.0.2 (Lollipop). The results were identical to the tests ran with NetworkManager, with the exception that the list of inner authentication methods of PEAP were reduced to MSCHAPv2 and GTC.

3.2 DES key cracking

Once the MSCHAPv2 handshake is retrieved and the response split into the three ciphertexts, the three DES keys can be retrieved by brute force and concatenated

to retrieve the users NT hash. The first step in doing so is to crack the third ciphertext. The reason for this is that since its key space is so small, it is quicker to crack it separately rather than waste resources on it while cracking keys one and two. Moxie Marlinspike has released a tool named "chapcrack" which splits the MSCHAPv2 response into three ciphertexts and cracks the third key as well. Its main use however is to generate a submission for cloudcracker, which claims to be able to retrieve the DES keys in under 21 hours for the price of 100 dollars [Int16]. Cracking DES is also feasible with the amount of computing power available in modern GPU's. Descrack, a DES cracker utilizing the OpenCL framework for cross-platform parallel programming, was able to achieve a rate of 3610.83 million keys per second on an AMD R9 270x graphics card [DT, Gro]. This means that the entire DES key space can be searched through in 230 days with a graphics card that is currently with 150 euros [Ord]. For the output of Descrack being ran on said graphics card, see the appendix. Some modifications would have to be made to the Descrack utility to be able to be able to search the two keys during one iteration. However, it is clear that an attacker gaining the user's NT hash by finding the three DES keys by brute force is a realistic possibility.

3.3 Authenticating to Samba

Samba is an open-source software suite designed to provide users file and printer sharing by implementing the Server Message Block protocol [Tea]. Tartu University utilizes Samba to share printers, synchronize users' home folders across, and to provide access to the files of the faculty's website and the computer science institute's clerical documents. For a list of publicly visible items shared on one of the university's servers, see the appendix. Tartu University's Samba server supports NTLM authentication, which means it is possible for an attacker to authenticate as the user once he has retrieved the user's NT hash. This authentication is greatly simplified by the fact that smbclient, a command-line utility provided by the Samba team, supports a `-pw-nt-hash` option, which allows users to authenticate while substituting their password with the NT hash [Tea13]. For example, the following command can be used by a computer science student in the University of Tartu to access their school computers' home directory:

```
smbclient //math.ut.ee/<username> -U <username>%<nt-hash>  
--pw-nt-hash -W DOMENIS
```

Therefore, once the attacker has a user's NT hash, gaining access to files for which they have permission in the school Samba file share is trivial.

3.4 Potential improvements for eduroam and UT

This section discusses different methods for improving the security of the current eduroam configuration and to mitigate the effects of the aforementioned attack.

Disable NTLM authentication for Samba The simplest option would indeed be to disable NTLM authentication for Samba. However, this still leaves the NT hash available to the attacker. The University of Tartu uses other services which also provide NTLM authentication such as OneDrive [Mic]. Disabling NTLM authentication in every service could potentially be cumbersome and it is possible that human error would lead to NTLM being enabled in a future service. In addition, access to the NT hash opens up the possibility of an offline dictionary attack, which would then reveal the user's password and in turn lead to even more serious consequences. The threat of a dictionary attack is increased by the fact that the NT hash operation is quite fast - using an AMD R9 290x graphics card, the time it takes to perform 100 trillion (10^{14}) guesses is 6 hours [Cra15]. For comparison, it takes 3 weeks to perform the same amount of operations on the same hardware using SHA256 and 904 years using SHA512crypt.

Validating the server certificate It is true that once server the server certificate is validated the user will not initiate the MS-CHAPv2 exchange with a rogue access point. This means that the attacker will not even get the required information to gain access to the user's NT hash. However, as there is no way for the authenticator to force the client to verify its certificate, this approach requires a lot of effort from the users. Each user would be responsible for setting up the certificate verification on their device and thus it is highly unlikely that this change would be adopted by the entire university. Furthermore, even when server certificate verification is enabled, it is possible that the users may choose to ignore warnings of a certificate which does not match. The change in certificate can be seen as a certificate update or a configuration mistake by the network administrators. Some users may not understand the warning at all. Users wish to connect to the internet quickly and they will most likely be willing to dismiss a warning if it means they might be able to connect.

Making eduroam passwords independent from UT passwords Gaining access to UT services would not be possible with this attack if UT passwords and eduroam passwords were separated and policies would be enforced which require them to differ from one another. A separate eduroam password would also provide protection against attacks which aim to retrieve the eduroam password as it is stored on the device. While an attacker could still perform a Man-in-the-Middle attack once they had the NT hash of the user, many sites that contain sensitive

information, such as ÕIS and moodle, use encryption, which means an attacker would most likely be unable to retrieve anything truly important. An attacker could also then perform a dictionary attack against the NT hash of the eduroam password in the hopes that the target uses the same base word for the UT password with a different number appended to the end, which is a common way to get around password policy requirements [Tru12]. That, however, would be an extraordinary amount of effort for a fairly small chance of a reward, making it fairly unlikely. This approach potentially requires a great deal of effort by the university's staff as implementing this solution might come with some changes in infrastructure.

Choose a different EAP method Switching to a more secure EAP method and eliminating the use of PEAP-MSCHAPv2 would definitely counter this attack effectively. However, finding a suitable replacement proves to be difficult. Of the older, well-supported EAP methods, the one that brings an increase in security over PEAP-MSCHAPv2 is EAP-TLS, a certificate-based authentication method [Hur08]. While EAP-TLS completely solves the root cause of this attack, the fact that eduroam uses the same username/password combination as the university's services, it also requires a client-side certificate. This would make eduroam a lot less convenient to use for the average user and thus widespread adoption could prove to be a problem. A new EAP method which provides security without a client-side certificate is EAP-pwd, also known as dragonfly, a password-authenticated key agreement scheme [Har12a]. EAP-pwd also provides resistance towards other attacks such as dictionary attacks and Man-in-the-Middle [Zor10]. EAP-pwd support is provided in Android 4.0, wpa_supplicant, hostapd, FreeRADIUS and Radiator [Har12b, McC12, Pro15].

While there is no simple solution to this attack which could be implemented with minimal effort, there are small improvements that can be made to improve the situation. While certificate verification can't be strictly enforced, the university's configuration guides should not instruct users to disregard the server certificate. The guide should definitely include information on how to retrieve and verify the server certificate. In addition, giving security-minded users the option to have an entirely separate eduroam password would be a strong deterrent against attackers trying to gain access to UT services via vulnerabilities in eduroam and would provide users with some peace of mind. The same principal applies to the ability to gain client certificates for EAP-TLS use.

4 Results

This aim of this research was to identify potential weaknesses in the University of Tartu's current eduroam configuration. During research, different attack vectors were tried until an attack was found in which an attacker can retrieve a user's NT hash. The specifics of this attack was then tested on different operating systems and under different conditions. All that is required of the user to be subject to this attack is to ignore a single certificate warning. For some users, their configuration has made it so that they forfeit that information as soon as they boot up their laptop and attempt to automatically connect.

The NT hash retrieved from the attack can be used to authenticate to the UT Samba server. It can also be used to perform a dictionary or brute-force attack. Therefore, a considerable security risk has been found in the university's eduroam configuration. This attack requires a response from the University of Tartu's network administrators. This need for a response is amplified by the fact that unless the university is willing to create separate passwords for each user's eduroam account or switch to certificate-based authentication there does not currently seem to be any solution which is able to shut down this attack completely.

This research was focused on the security of eduroam passwords during wireless authentication. Additional vulnerabilities could possibly be found by finding weaknesses in the ways that different operating systems and network configuration managers store these passwords.

5 Teenuse *eduroam* paroolide turvalisus

Bakalaureusetöö(9 EAP)
Raul-Martin Rebane

Selle bakalaureustöö eesmärk oli leida võimalikke nõrkusi Tartu Ülikooli *eduroam* konfiguratsioonis. Uuringu käigus prooviti erinevaid rünnakuvektoreid kuni leidis rünnak, mille abil oli ründajal võimalik kätte saada kasutaja NT räsi. Selle rünnaku üksikasju seejärel testiti erinevate operatsioonisüsteemide all erinevates olukordades. Kõik, mis on praegusel hetkel vaja selleks, et kasutaja langeks selle rünnaku ohvriks, on see et kasutaja eirab ühte sertifikaadihoiatust. Mõnede kasutajate puhul loovutavad nad ründajale vajaliku info vaid sellega, et käivitavad oma sülearvuti, mis seejärel üritab automaatselt sertifikaati kasutamata *eduroami* ühendada.

Rünnaku jooksul kätte saadud kasutaja NT räsi saab seejärel kasutada Tartu Ülikooli Samba serveriga autentimiseks. Seda saab lisaks kasutada ka sõnaraamatu rünnakute või rünnakuteks, kus proovitakse läbi kõik võimalikud tähekombinatsioonid. Seega on ülikooli *eduroami* häälestusest leitud arvestatav turvarisk, mis nõuab Tartu Ülikooli võrguadministraatorite tähelepanu. Antud turvariskile ei tundu ka hetkel leiduvat lahendust, mis ei nõua teenuse *eduroam* paroolide eraldamist Tartu Ülikooli kontro paroolidest või vahetust kliendipoolseid sertifikaate nõudvale autentimismeetodile.

Antud uurimistöö keskendus teenuse *eduroam* paroolide turvalisusele traadita ühenduse autentimishetkel. On võimalik, et leiduvad teised nõrkused, mis keskenduvad sellele, kuidas erinevad operatsioonid või võrgukonfiguratsiooni haldajad talletavad neid paroole.

References

- [Abo05] D. Stanley J. Walker B. Aboba. Extensible Authentication Protocol (EAP) Method Requirements for Wireless LANs. IETF Request for Comments: 4017, March 2005. <https://www.ietf.org/rfc/rfc4017.txt>.
- [Ant] Joshua Wright Brad Antoniewicz. FreeRADIUS-WPE. http://www.willhackforsushi.com/?page_id=37.
- [Aut15] Internet Assigned Numbers Authority. Extensible Authentication Protocol (EAP) Registry, February 2015. <https://www.iana.org/assignments/eap-numbers/eap-numbers.xhtml>.
- [Bur] William E. Burr. Data Encryption Standard. <http://nvlpubs.nist.gov/nistpubs/sp958-lide/250-253.pdf>.
- [CB12] Arran Cudbard-Bell. protocol EAP PEAP. FreeRADIUS wiki, September 2012. <http://wiki.freeradius.org/protocol/EAP-PEAP>.
- [Cob98] G. Zorn S. Cobb. Microsoft PPP CHAP Extensions. IETF Request for Comments: 2433, October 1998. <https://tools.ietf.org/html/rfc2433>.
- [Cor15] Microsoft Corporation. [MS-NLMP]: NT LAN Manager (NTLM) Authentication Protocol, October 2015. <http://download.microsoft.com/download/9/5/E/95EF66AF-9026-4BB0-A41D-A4F81802D92C/%5BMS-NLMP%5D.pdf>.
- [Cra15] Blase Ur Fumiko Noma Jonathan Bees Sean M. Segreti Richard Shay Lujo Bauer Nicolas Christin Lorrie Faith Cranor. "I Added '!' at the End to Make It Secure": Observing Password Creation in the Lab, July 2015. <https://www.usenix.org/system/files/conference/soups2015/soups15-paper-ur.pdf>.
- [DT] Danilo Barga Daniel Thornburgh. Descrack. Github repository. <https://github.com/dbrgn/descrack>.
- [Fou98a] Electronic Frontier Foundation. Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design, May 1998. <http://cryptome.org/jya/cracking-des/cracking-des.htm>.
- [Fou98b] Electronic Frontier Foundation. Frequently Asked Questions (FAQ) About the Electronic Frontier Foundation's "DES Cracker" Machine, July 1998. https://w2.eff.org/Privacy/Crypto/Crypto_misc/DESCracker/HTML/19980716_eff_des_faq.html.

- [Gro] Khronos Group. OpenCL. <https://www.khronos.org/opencv/>.
- [gro08] S. Winter T. Kersting P. Dekkers L. Guido S. Papageorgiou Janos Mohacsi R. Papez M. Milinovic D. Penezic J. Rauschenbach J. Tomasek K. Wierenga T. Wolniewicz J.-M. Macias-Luna I. Thomson JRA5 group. Inter-NREN Roaming Infrastructure and Service Support Cookbook - Third Edition, October 2008. <https://www.eduroam.org/downloads/docs/GN2-08-230-DJ5.1.5.3-eduroamCookbook.pdf>.
- [Har12a] Dan Harkins. Dragonfly: A PAKE scheme. IETF 83 Proceedings, March 2012. <https://www.ietf.org/proceedings/83/slides/slides-83-cfrg-0.pdf>.
- [Har12b] Daniel Harkins. Secure authentication with only a password, May 2012. <http://community.arubanetworks.com/t5/Technology-Blog/Secure-authentication-with-only-a-password/ba-p/36524>.
- [HK97] R. Canetti H. Krawczyk, M. Bellare. HMAC: Keyed-Hashing for Message Authentication. IETF Request for Comments: 2104, February 1997. <http://www.ietf.org/rfc/rfc2104.txt>.
- [Hur08] D. Simon B. Aboba R. Hurst. The EAP-TLS Authentication Protocol. IETF Request for Comments: 5216, March 2008. <https://tools.ietf.org/html/rfc5216>.
- [Int] International Business Machines Corporation (IBM). IBM Archives: 1970s. http://www-03.ibm.com/ibm/history/history/year_1977.html.
- [Int16] Internet Archive: Wayback Machine. Cloudcracker website, March 2016. <https://web.archive.org/web/20160319080738/https://www.cloudcracker.com/>.
- [Joh98] Thomas R. Johnson. American Cryptology during the Cold war, 1945-1989, February 1998. <http://nsarchive.gwu.edu/NSAEBB/NSAEBB260/nsa-6.pdf> pages 231-233.
- [Jos03] Ashwin Palekar Dan Simon Glen Zorn S. Josefsson. Protected EAP Protocol (PEAP). IETF Internet-Draft, October 2003. <https://tools.ietf.org/html/draft-josefsson-pppext-eap-tls-eap-06>.
- [Kal16a] Roman Kaljuorg. Connecting to eduroam (Android), February 2016. <https://wiki.ut.ee/display/AA/Android>.

- [Kal16b] Roman Kaljuorg. Connecting to eduroam (Windows 8 & 10), February 2016. <https://wiki.ut.ee/pages/viewpage.action?pageId=33227669>.
- [Lev04] Bernard Aboba Larry J. Blunk John R. Vollbrecht James Carlson Henrik Levkowetz. Extensible Authentication Protocol. IETF Request for Comments: 3748, June 2004. <https://tools.ietf.org/html/rfc3748>.
- [Mar12] Moxie Marlinspike. Divide and Conquer: Cracking MS-CHAPv2 with a 100via Internet Archive: Wayback Machine, July 2012. <https://web.archive.org/web/20160316174007/https://www.cloudcracker.com/blog/2012/07/29/cracking-ms-chap-v2/>.
- [MC98] J. Dolske M. Curtin. A Brute Force Search of DES Keyspace, April 1998. <http://www.interhack.net/pubs/des-key-crack/>.
- [McC12] Mike McCauley. Added support for EAP-PWD per RFC 5931. mailing list announcement, June 2012. <http://www.open.com.au/pipermail/radiator-announce/2012-June/000018.html>.
- [Mic] Microsoft. Use the OneDrive for Business app on an iphone or ipad. <https://support.office.com/en-us/article/Use-the-OneDrive-for-Business-app-on-an-iPhone-or-iPad%a1dbdcda-19df-4c06-b908-29c10865d20d>.
- [Mic08] Microsoft. Extensible Authentication Protocol overview, February 2008. <https://technet.microsoft.com/en-us/network/cc917480>.
- [Mic15] Microsoft Corporation. [MS-PEAP]: Protected Extensible Authentication Protocol (PEAP), October 2015. <http://download.microsoft.com/download/9/5/e/95ef66af-9026-4bb0-a41d-a4f81802d92c/%5BMS-PEAP%5D.pdf>.
- [MM12] members of the SA3 T2 group M. Milinović, Stefan Winter. eduroam Policy Service Definition Version 2.8, July 2012. https://www.eduroam.org/downloads/docs/GN3-12-192_eduroam-policy-service-definition_ver28_26072012.pdf.
- [Net] Aruba Networks. A Tour of the EAP-PEAP-MSCHAPv2 Ladder. http://www.arubanetworks.com/techdocs/ClearPass/Aruba_DeployGd_HTML/Content/A%20802.1X%20EAP-PEAP-AD%20Reference/EAP_PEAP_handshake.htm.

- [Ord] AS Ordi. 2GB MSI R9 270 Gaming PCI-e. <http://www.ordi.ee/EPood/Product.aspx?MC=KOMP&IC=1700&PGC=PCIE&ItemID=1700-2275>.
- [oST] National Institute of Standards and Technology. Archived FIPS publications. <http://csrc.nist.gov/publications/PubsFIPSArch.html>.
- [Pro] The FreeRADIUS Project. About the FreeRADIUS Project. <http://freeradius.org/about.html>.
- [Pro15] The FreeRADIUS Project. FreeRADIUS source code, May 2015. http://code.metager.de/source/xref/freeradius/server/src/modules/rlm_eap/types/rlm_eap_pwd/.
- [Rei16] R. Van Rein. Kerberos in the Extensible Authentication Protocol (EAP-Kerberos). IETF Internet-Draft, March 2016. https://datatracker.ietf.org/doc/draft-vanrein-eap-kerberos/?include_text=1.
- [RK16] Heiki Int Roman Kaljuorg. Connecting to eduroam (Windows 7), February 2016. <https://wiki.ut.ee/display/AA/Windows+7>.
- [Sim96] W. Simpson. PPP Challenge Handshake Authentication Protocol (CHAP). IETF Request for Comments: 1994, August 1996. <https://tools.ietf.org/html/rfc1994>.
- [Sys] Lancom Systems. PEAP-Default-Tunnel-Method. Online. https://www.lancom-systems.de/docs/LCOS-Menu/9.10-Rel/EN/topics/2_25_10_10_6.html.
- [Sys08] Cisco Systems. Wi-Fi Protected Access 2 (WPA 2) Configuration Example, January 2008. <http://www.cisco.com/c/en/us/support/docs/wireless-mobility/wireless-lan-wlan/67134-wpa2-config.html>.
- [Tea] The Samba Team. What is samba? https://www.samba.org/samba/what_is_samba.html.
- [Tea13] The Samba Team. smbclient. Samba man pages, May 2013. <https://www.samba.org/samba/docs/man/manpages-3/smbclient.1.html>.
- [The16] The NetworkManager Team. NetworkManager, April 2016. <https://wiki.gnome.org/Projects/NetworkManager>.
- [Tru12] Trustwave. 2012 global security report, 2012. <https://www.trustwave.com/Resources/Library/Documents/2012-Trustwave-Global-Security-Report/?dl=1>.

- [Zor00] G. Zorn. Microsoft PPP CHAP Extensions, Version 2. IETF Request for Comments: 2759, January 2000. <https://tools.ietf.org/html/rfc2759>.
- [Zor10] D. Harkins G. Zorn. Extensible Authentication Protocol (EAP) Authentication Using Only a Password. IETF Request for Comments: 5931, August 2010. <https://tools.ietf.org/html/rfc5931>.

All links were valid on 12.05.2016.

Appendices

1 NetworkManager eduroam configuration

```
[ connection ]
id=eduroam
uuid=51cfb60f 9dba 4191 882c 8aa000122b6b
type=wifi
permissions=
secondaries=

[ wifi ]
mac_address=B4:B6:76:93:BF:0C
mac_address_blacklist=
mode=infrastructure
seen_bssids=
ssid=eduroam

[ wifi security ]
auth_alg=open
group=
key_mgmt=wpa eap
pairwise=
proto=

[ 802 1x ]
altsubject_matches=
eap=ttls ;
identity=<ut username>
password=<ut password>
phase2_altsubject_matches=
phase2_auth_eap=mschapv2

[ ipv4 ]
dns_search=
method=auto

[ ipv6 ]
dns_search=
method=auto
```

2 Descracker output

[illegible]

3 math.ut.ee Samba shares

| Domain=[DOMENIS] OS=[Unix] Server=[Samba 4.0.26 SerNet RedHat 8.el6] | | |
|--|---------|---|
| Sharename | Type | Comment |
| test | Disk | kontrollin... |
| netlogon | Disk | NetLogon service Samba 4.0.26 SerNet |
| RedHat 8.el6 | | |
| materjalid | Disk | Materjalid tudengitele |
| atidoc | Disk | ATI dokumendid |
| atidoc217 | Disk | ATI dokumendid |
| atiproj | Disk | ATI projektijuhid |
| ati_pop | Disk | ATI populariseerimine |
| mmidoc | Disk | MMI dokumendid |
| mmi_avalik | Disk | MMI avalik |
| dekdoc | Disk | Dekanaadi dokumendid |
| dek_avalik | Disk | Dekanaadi dokumendid levitamiseks |
| rmtkdoc | Disk | Raamatukogu dokumendid |
| samba | Disk | Samba directory. |
| www | Disk | www.cs.ut.ee data files. |
| msiweb | Disk | MSI veeb |
| rmiweb | Disk | RMI veeb |
| mathweb | Disk | math.ut.ee veeb (mitte MT veeb!) |
| msi_avalik | Disk | MSI dokumendid |
| msi_kantselei | Disk | MSI kantselei dokumendid |
| msiskanner | Disk | |
| profiles | Disk | user profiles |
| programs | Disk | Programmid |
| koolitus | Disk | Koolitus |
| videod | Disk | Maria videoarhiiv |
| enobollo | Disk | Eno share bollos |
| IPC\$ | IPC | IPC Service (Samba 4.0.26 SerNet RedHat |
| 8.el6) | | |
| rmtkprint | Printer | MT raamatukogu HP LaserJet 2055dn |
| mmiprint | Printer | MMI HP LaserJet P2055dn |
| hplaser2 | Printer | ATI HP LaserJet P2015dn |
| hplaser3 | Printer | ATI HP LaserJet P2015x |
| dkprint | Printer | Dekanaadi HP LaserJet 400 M401dne |
| HP Officejet Pro 17700 | Printer | HP Officejet Pro dekanadis |
| mtat hplaser400 | Printer | MTAT kantselei HP LaserJet 400 MFP M425dn |
| biit 1 | Printer | ATI BIIT HP LaserJet P2015dn |
| biit 2 | Printer | ATI BIIT HP LaserJet P2055dn |
| mtat hpcolor | Printer | ATI mtat hpcolor HP LaserJet 500 color |
| M551 | | |
| hplaser | Printer | ATI HP LaserJet P4015dn |
| msiprint | Printer | MSI Ricoh MP3350 |
| dkprint 2 | Printer | Dekanaadi HP LaserJet P2055dn (2) |

| | | |
|--|----------------------------------|---|
| itocanon | Printer | ITO Canon LBP 2460 |
| pmiprint | Printer | HP LaserJet MFP M630 |
| hp2420 | Printer | hp2420 |
| mtms hpcolor | Printer | mtms hpcolor HP LaserJet 500 color M551 |
| r005 | Printer | HP LaserJet P2015 DN |
| r204 | Printer | Xerox ColorQube 8570 |
| mtmt hpojcolor | Printer | MIMT HP Officejet Pro 2. korrusel |
| csproj | Printer | ATI 335 printer HP LaserJet 400 M401dne |
| mmicolor | Printer | MMI HP Color LaserJet CP4025dn |
| rmiprint | Printer | MT raamatukogu HP LaserJet 2055dn |
| mmiprint2 | Printer | MMI Samsung 2015 |
| aticanon | Printer | ATI Canon iR3225N |
| Domain=[DOMENIS] OS=[Unix] Server=[Samba 4.0.26 SerNet RedHat 8.el6] | | |
| Server | Comment | |
| NEWMATH | Samba 4.0.26 SerNet RedHat 8.el6 | |
| Workgroup | Master | |
| DOMENIS | | |

Non-exclusive licence to reproduce thesis and make thesis public

I, Raul-Martin Rebane (date of birth: 6th of March 1993),

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

1.1 reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

1.2 make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

Security of eduroam passwords

supervised by Dominique Unruh

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, 12.05.2016