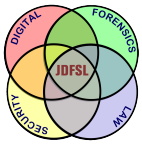

THE JOURNAL OF
DIGITAL FORENSICS,
SECURITY AND LAW

Journal of Digital Forensics, Security and Law

Volume 13 | Number 1 Article 9

3-31-2018

Automated Man-in-the-Middle Attack Against Wi-Fi Networks

Martin Vondráček
Brno University of Technology, Brno, Czech Republic, xvondr20@stud.fit.vutbr.cz

Jan Pluskal
Brno University of Technology, Brno, Czech Republic, ipluskal@fit.vutbr.cz

Ondřej Ryšavý
Brno University of Technology, Brno, Czech Republic, rysavy@fit.vutbr.cz

Automated Man-in-the-Middle Attack Against Wi-Fi Networks

Cover Page Footnote
This paper is an extended version of the original paper that has been presented at the 9th EAI International Conference on Digital Forensics and Cyber Crime (Vondráček, Pluskal, & Ryšavý, 2018). This work was supported by Ministry of Interior of the Czech Republic project "Integrated platform for analysis of digital data from security incidents" VI20172020062; Ministry of Education, Youth and Sports of the Czech Republic from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science" LQ1602; and by BUT internal project "ICT tools, methods and technologies for smart cities" FIT-S-17-3964.

AUTOMATED MAN-IN-THE-MIDDLE ATTACK AGAINST WI-FI NETWORKS

Martin Vondráček Jan Pluskal Ondřej Ryšavý
Brno University of Technology
Faculty of Information Technology
Božetěchova 2, Brno, Czech Republic
{xvondr20}@stud.fit.vutbr.cz, {ipluskal,rysavy}@fit.vutbr.cz

ABSTRACT

Currently used wireless communication technologies suffer security weaknesses that can be exploited allowing to eavesdrop or to spoof network communication. In this paper, we present a practical tool that can automate the attack on wireless security. The developed package called *wifimitm* provides functionality for the automation of *MitM* attacks in the wireless environment. The package combines several existing tools and attack strategies to bypass the wireless security mechanisms, such as WEP, WPA, and WPS. The presented tool can be integrated into a solution for automated penetration testing. Also, a popularization of the fact that such attacks can be easily automated should raise public awareness about the state of wireless security.

Keywords: Man-in-the-Middle attack, accessing secured wireless networks, password cracking, dictionary personalization, tampering network topology, impersonation, phishing

1. INTRODUCTION

Recent enhancements to wireless technology strengthen the benefits of wireless communication. It is convenient to access the network from any location within the network coverage area. For most of the portable devices, this is the only way to connect to the network. Installation and network setup are easy, and the network is further expandable. The main benefit of Wi-Fi, its accessibility, makes this technology a suitable target of attacks. A potential attacker needs to be in the physical proximity of a Wi-Fi network. The proposed wireless security standards aim at prevention of such unauthorized access. Unfortunately, the first standard called WEP is so weak that it is possible to crack the password in a few seconds using a conventional laptop computer. The answer was the introduction of stronger standard WPA and later even stronger WPA2. In 2017, Mathy Vanhoef announced that he discovered a vulnerability in security mechanisms that use the four-way handshake (WPA and WPA2) and demonstrated how easily this vulnerability can be exploited.

The main focus of this paper is security of wireless networks. It provides a study of widely used network technologies and mechanisms of wireless security. Analyzed tech-

[1]This paper is an extended version of the original paper that has been presented at the 9th EAI International Conference on Digital Forensics and Cyber Crime (Vondráček, Pluskal, & Ryšavý, 2018).

nologies and security algorithms suffer weaknesses that can be exploited to perform Man-in-the-Middle attacks. A successful realization of this kind of attack allows not only to eavesdrop on all the victim's network traffic but also to spoof his communication (Prowell, Kraus, & Borkin, 2010, pp. 101–120; Callegati, Cerroni, & Ramilli, 2009).

In an example scenario (Figure 1), the victim is a suspect conducting illegal activity on a target network. The attacker is a law-enforcement agency investigator with appropriate legal authorization to intercept the suspect's communication and to perform a direct attack on the network. In some cases, the suspect may be aware that his communication can be intercepted by the Internet Service Provider and harden his network. For example, he could use an overlay network technology, e.g., *VPN* (implemented by *L2TP*, *IPsec* (Kent & Seo, 2005, pp. 09–10), *PPTP*) or anonymization networks (Tor, I2P, etc.) to create an encrypted tunnel configured on his gateway, for all his external communication. This concept is easy to implement and does not require any additional configuration on endpoint devices. Generally, this would not be considered a properly secured network (Godber & Dasgupta, 2003, pp. 425-431), but this scheme, or similar, is often used by large vendors like Cisco (Deal & Cisco Systems, 2006) or Microsoft (Thomas, 2017) for branch office deployment and can also be seen in home routers[1]. In such cases, intercepting traffic on the ISP level would not yield meaningful results, because all the communication is encrypted by the hardening. On the other hand, direct attack on the suspect's LAN will intercept plain communication. But, even when an investigator is legally permitted to carry out such an attack to acquire evidence, it is scarcely used, because it requires expert domain knowledge. Thus, this process of evidence collection is very expensive and human resource demanding.

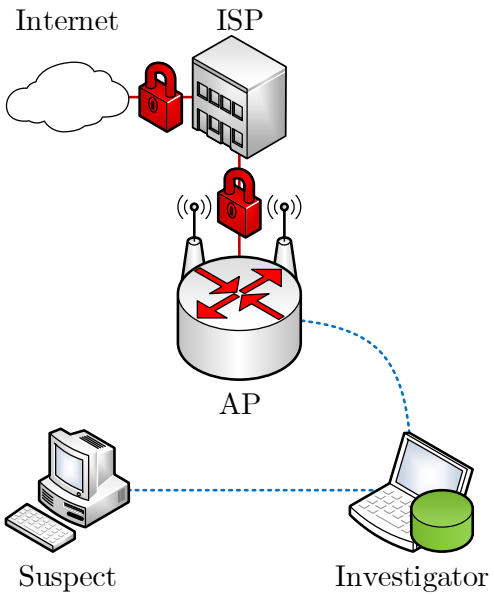

Figure 1. Example forensics scenario where the suspect has hardened his network and uses an encrypted tunnel from the gateway (*AP*).

The aim of this research is to design, implement and test a tool able to automate the process of accessing a secured *WLAN* and to perform data interception. Furthermore, this tool should be able to tamper with the network to collect more evidence by redirecting traffic to place itself in the middle of the communication and tamper with it, to access otherwise encrypted data in plain form. Using the automated tool should not require any expert knowledge from the investigator.

We designed a generic framework, see Figure 3, capable of accessing and acquiring evidence from a wireless network regardless of used security mechanisms. This framework can be split into several steps. First,

[1]Asus RT-AC5300 – Merlin WRT has an option to tunnel all traffic thought Tor.

it is necessary for an investigator to obtain access to the *WLAN* used by the suspect. Therefore, this research focuses on exploitable weaknesses of particular security mechanisms, see Section 2 for more details. Upon successful connection to the network, the investigator needs to tamper with the network topology. For this purpose, weaknesses of several network technologies can be exploited. From this point on, the investigator can start to capture and break the encryption on the suspect's communication.

Specialized tools focused on exploiting individual weaknesses in security mechanisms currently used by *WLAN*s are already available. There are also specialized tools focused on individual steps of *MitM* attacks. Tools that were analyzed and used in implementation of the *wifimitm* package are outlined in Section 2.

Based on the acquired knowledge, referenced studies and practical experience from manual experiments, authors were able to create an attack strategy which is composed of a suitable set of available tools. The strategy is then able to select and manage individual steps for a successful *MitM* attack tailored to a specific *WLAN* configuration. This strategy also includes options for impersonation and phishing for situations, when the network is properly secured, and the weakest part of the overall security is the suspect.

The created software can perform a fully automated attack and requires zero knowledge. We tested the implementation on carefully devised experiments, with available equipment. The tool is open source and can be easily incorporated into other software. The main use cases of this tool are found in automated penetration testing, forensic investigation, and education.

2. SECURITY WEAKNESSES IN WLAN TECHNOLOGIES

Following network technologies (Sections 2.1, 2.2), which find a significant utilization, unfortunately, suffer from security weaknesses in their protocols. These flaws can be used in the process of the *MitM* attack.

2.1 Wireless Security

Wired Equivalent Privacy (*WEP*) is a security algorithm introduced as a part of the IEEE 802.11 standard (Halsall, 2005, p. 665; IEEE-SA, 2012, pp. 1167–1169). Today, *WEP* is deprecated and superseded by subsequent algorithms, but is still sometimes used, as can be seen from Table 1 available from *Wifileaks.cz*[2]. Fluhrer, Mantin, and Shamir (2001) presented that *WEP* is broken. There are tools that provide access to wireless networks secured by *WEP* available (Tews, Weinmann, & Pyshkin, 2007). Regarding *WEP* secured *WLAN*s, authentication can be either *Open System Authentication* (*OSA*) or *Shared Key Authentication* (*SKA*) (IEEE-SA, 2012, pp. 1170–1174). In the case of *WEP OSA*, any *station* (*STA*) can successfully authenticate to the *Access Point* (*AP*) (Robyns, 2014, pp. 4–10). *WEP SKA* provides authentication and security of transferred communication using a shared key. Confidentiality of transferred data is ensured by encryption using the *RC4* stream cipher. Methods used for cracking access to *WEP* secured networks are based on analysis of transferred data with corresponding *Initialization Vectors* (*IV*s).

Wi-Fi Protected Access® (*WPA*™, a subset of 802.11i) was developed by the Wi-Fi Alliance® as a reaction to increasing number of security flaws in *WEP*. The *WPA* is de-

[2] `http://www.wifileaks.cz/statistika/`

signed to be backward hardware compatible with devices that used *WEP*, and vendors were expected to provide a firmware update to remedy the catastrophic situation with *WEP*. Therefore, for data confidentiality and integrity was chosen *Temporal Key Integrity Protocol* (*TKIP*). The main flaw of *WPA* security algorithm is associated with the *TKIP* and a four-way handshake. It can be identified at the beginning of client device's communication, where an unsecured exchange of confidential information is performed during the handshake. An investigator can obtain this unsecured communication and use it for consecutive cracking of the *Pairwise Master Key* (*PMK*) that is derived from *Pre-Shared Key* (*PSK*) or negotiated using an 802.1x authentication stage in case of enterprise authentication.

Wi-Fi Protected Access® 2 (*WPA2™*, full implementation of 802.11i) is a successor of *WPA*, but security flaws of the *WPA* algorithm remain significant also for the *WPA2*. Besides *TKIP*, *WPA2* has mandatory support of *Counter Mode CBC-MAC Protocol* (*CCMP*). Both *TKIP* and *CCMP* ensure data confidentiality, authentication, and access control. IEEE 802.11ad adds and 802.11ac extends a new confidentiality protocol *Galois/Counter Mode Protocol* (*GCMP*). Information exposed during the handshake can be once again used for the dictionary attack, which can be further improved by precomputing the *PMKs* (Kumkar, Tiwari, Tiwari, Gupta, & Shrawne, 2012, pp. 37–38; Liu, Jin, & Wang, 2010, p. 3). Precomputed lookup tables are already available online[3].

A critical security flaw in wireless networks secured by *WPA* or *WPA2* is the functionality called *Wi-Fi Protected Setup™* (*WPS*). This technology provides a comfort-

[3] `https://www.renderlab.net/projects/WPA-tables/`

Table 1. Following table summarizes *WLAN* statistics provided by *Wifileaks.cz*. Users of this service voluntarily scan and publish details about *WLAN*s in the Czech Republic. Information in the table show that a significant number of *WLAN*s still use deprecated security algorithms. The statistics consisting of 97 192 922 measurements of 2 548 054 *WLAN*s were published on May 26, 2017.

Security	Count	Ratio
WPA2	1 429 518	56 %
WEP	393 579	15 %
WPA	375 984	15 %
open	67 388	3 %
other	281 585	11 %

Table 2. Results of wardriving in Bratislava and Brno focused on UPC vulnerabilities concerning default *WPA2 PSK* passwords (Klinec & Svítok, 2016b). Detailed article about these security flaws is available online (Klinec & Svítok, 2016a).

Bratislava, Slovakia, 2016-10-01	Count	Ratio
Total networks	22 172	
UPC networks	3 092	13.95 %
Vulnerable UPC networks	1 327	42.92 % UPC
Brno, Czech Republic, 2016-02-10	**Count**	**Ratio**
Total networks	17 516	
UPC networks	2 868	16.37 %
Vulnerable UPC networks	1 835	63.98 % UPC

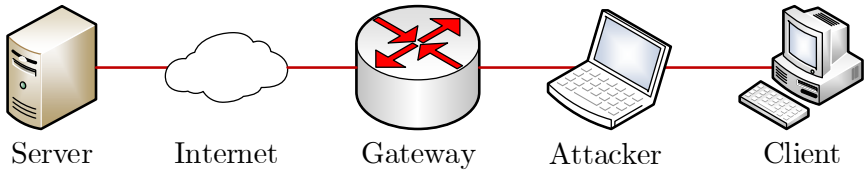

Figure 2. In an example network topology suitable for realization of *MitM* attack, the attacker's device acts towards the victim as a default gateway. All the communication routed outside the local network from the victim is sent to the default gateway, in this case to the attacker's device. From the attacker's device, the communication can be further routed to the real default gateway (Callegati et al., 2009). For the successful execution of this scenario, the attacker needs to be connected to the targeted local network.

able and supposedly secure way of connecting to the network. For a connection to the *WLAN* with *WPS* enabled, it is possible to use an individual *PIN*. However, the process of connecting to the properly secured network by providing *PIN* is very prone to brute-force attacks (Heffner, 2011). Because *WPS* is a usual feature in today's access points and that *WPS* is usually turned on by default, *WPS* can be a very common security flaw even in networks secured by *WPA2* with a strong password. Currently, there are already available automated tools for exploiting *WPS* weaknesses, e.g., *Reaver Open Source*[4].

Recently, a critical vulnerability, *Key Reinstallation Attacks* (*KRACKs*), was discovered by Vanhoef & Piessens, 2017 revealing a flaw in 801.11i and related specifications, more precisely, in the description of the four-way handshake. A security of *CCMP* and *GCMP* encryption methods expects that no *Initialization Vector* (*IV*) repeats under the same key. Authors showed that abusing this vulnerability, they can reinstall a *Pairwise Transient Key* (*PTK*) used for generation of *Key Confirmation Key* (*KCK*), *Key Encryption Key* (*KEK*), and *Temporal Key* (*TK*). *KCK* and *KEK* are used for handshake protection and *TK* for data encryption. The reinstallation resets the *incremental transmit packet number* (nonce) and *receiver packet number* (replay counter) to the initial value. Therefore, the reinstallation violates the expectation of non reusable *IV*, which consequently breaks *TKIP*, *CCMP* or *GCMP* protocols. As Vanhoef & Piessens, 2017 show, this also occasionally happens in regular conditions, without an adversary.

Newly purchased access points usually use *WPA2* security by default. Currently, many access points can be found using default passwords not only for wireless network access, but even for *AP*'s web administration. With access to the *AP*'s administration, the investigator could focus on changing the network topology by tampering the network configuration. Access to the network management further allows the investigator to lower security levels, disable attack detections, reconfigure *DHCP* together with *DNS* and also clear *AP*'s logs. There are already implemented tools, which exploit relations between *SSID*s and default network passwords, e.g., *upc_keys*[5] by Peter Geissler.[6]

[4] `https://code.google.com/archive/p/reaver-wps/`

[5] `https://haxx.in/upc-wifi/`

[6] UPC company is a major ISP in the Czech Republic, URL: `https://www.upc.cz`

These tools could be used in an attack on the network with default *SSID* to improve dictionary attack using possible passwords. High severity of these security flaws is also proven by the fact that a significant amount of *WLAN*s was found using unchanged passwords, as it is shown in Table 2.

2.2 Network Technologies vulnerable to MitM

Man-in-the-middle attacks are possible because of the very nature of existing network protocols. No designed for providing security per see, the common network protocols lack strong authentication capabilities that would prevent their misuse by an attacker. Man-in-the-middle attacks assume that the attacker can divert legitimate communication. Switched Ethernet and secured wireless transmission separates the communication between two endpoints thus no other device should be able to see the conversation. Fortunately for the attacker, the insecurity of existing widely deployed protocols can be used. At the minimum, the following protocols can be considered as suitable targets:

1. *DHCP* automates network device configuration without a user's intervention (Droms, 1997).

2. *ARP* translates an *IPv4* address to a destination *MAC* address of the next-hop device in the local area network (Plummer, 1982).

3. *IPv6* networks utilize *ICMPv6 Neighbor Discovery* functionality to achieve similar functionality to *ARP* in *IPv4* networks.

Because of the lack of authentication and integrity checking, these protocols are vulnerable to spoofing attacks:

1. *DHCP Spoofing* generates fake *DHCP* communication. This attack can also be referred to as *Rogue DHCP*. An investigator can perform this kind of attack to provide devices in the network with malicious configuration, most often a fake default gateway address or *DNS* address.

2. *ARP Spoofing* provides the network devices with fake *ARP* messages. This persuades the suspect's device to believe that the attacking device's *MAC* address is the default gateway's *MAC* address.

3. *IPv6 Neighbor Spoofing* is a similar concept to *ARP Spoofing*.

From the available spoofing attacks, the *ARP Spoofing* technique was implemented in our tool. This method proved itself with reasonable performance during experiments and it is simple to implement.

Of course, there are counter-measures to spoofing attacks. The defense against spoofing lies in implementing some extra functionality to network devices:

1. *DHCP Snooping* is a countermeasure against *DHCP Spoofing*. This technique focuses on detection of forged *DHCP* communication. Network device acting as a *DHCP* snooper accepts only *DHCP* messages which are coming from connections to the genuine *DHCP* server, others are discarded. This way, individual connections are classified as either trusted or untrusted. If the network contains an unknown *DHCP* server behind an untrusted connection, it is referred to as *Spurious DHCP Server* (Cisco Systems, Inc., 2013, p. 54-2).

2. *Dynamic ARP Inspection* (*DAI*) is based on analysis of *ARP* messages

transmitted over the network with aim to detect *ARP Spoofing*. Similarly, device performing *DAI* can have its connections classified as trusted or untrusted. *ARP* messages from trusted connections are not checked. Analyzing device maintains a trusted database of mapping of *IP* and *MAC* addresses in the corresponding *LAN*. *ARP* messages from untrusted connections can be verified against this trusted mapping database (Cisco Systems, Inc., 2013, p. 56-2).

3. *Neighbor Discovery Inspection* (*NDI*) uses similar approach as above-mentioned *DAI*, but to detect *IPv6 Neighbor Spoofing*. Analyzing device verifies information transferred in Neighbor Discovery messages against its database of *IP* and *MAC* address mappings.

Although the mitigation techniques are known, they are applied mostly in the enterprise environments. In SOHO networks the devices either lack this feature or the protection is not enabled by the administrator.

2.3 Man-in-the-Middle Attack

The *MitM* refers to the situation, where the attacker's device is located in the network topology between two participants of the communication (Figure 2). The attacker then acts as an intermediary and the network traffic is routed through the attacking device. This state of unauthorized and intentionally changed network topology enables the attacker to eavesdrop on passed communication. The attacker is also able to focus on decryption of data and on changing the content of passed communication. That means that the attacker can inject harmful content. The attacker's prioritized intention is not only to take control over the traffic but also to perform this attack without anyone noticing it. This way, *Man-in-the-Middle Attack* is endangers maintaining confidentiality and integrity, key parts of the *CIA* triad.

HTTPS uses asymmetric cryptography with private and public keys to provide secure *HTTP* communication. If the victim is communicating using *HTTPS*, successful realization of *MitM* attack is more difficult. During communication of web browser on client's device with a web server, these two parties exchange a certificate containing a public key for providing a secure data transfer. *MitM* attack, in this case, captures transferred certificate and replaces it with a forged one (Callegati et al., 2009). The forged certificate is at this point a self-signed certificate. Upon reception of the self-signed certificate, victim's web browser can show some warning concerning possible risk. If the victim is not aware of the possible consequences, the victim can accept the certificate. In the case of success, both communicating devices are convinced of secured *HTTPS* communication, but the attacking device has the ongoing communication available.

DNS Spoofing focuses on possibilities of forging *DNS* communication used for resolution of domain names and *IP* addresses. For the successful realization of this attack, the attacker needs to detect and intercept *DNS* messages in the network. The aim of this attack is to direct the victim to a different device by providing a fake mapping of inquired domain name to a special *IP* address. The attacker is able to imitate the inquired service by running a similar rogue service on the provided spoofed *IP* address. If the victim is convinced that the inquired service is genuine, the attacker can then focus on capturing confidential information and credentials. The attacker can also use *DNS Spoofing* for providing the real service, but with

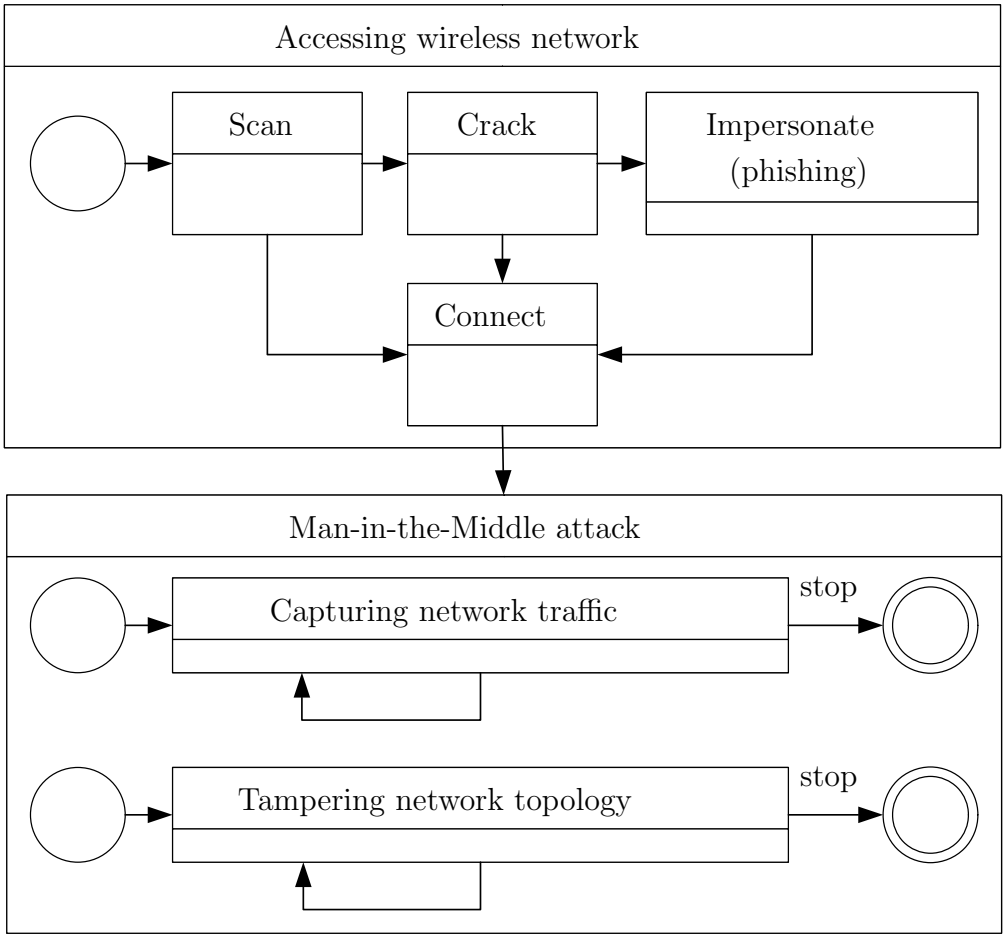

Figure 3. During the first phase – *Accessing wireless network*, the tool is capable of an attack on *WEP OSA*, *WEP SKA*, *WPA PSK* and *WPA2 PSK* secured *WLAN*s. In a case of the dictionary attack on the device deployed by the UPC company, used dictionaries are personalized by the implicit passwords. In the case of properly secured *WLAN*, impersonation (phishing) can be employed. Using this method, an investigator impersonates the legitimate network to obtain the *WLAN* credentials from the user. During the second phase – *Tampering network topology*, the tool needs to continuously work on keeping the network *stations* (*STA*s) persuaded that the spoofed topology is the correct one. An investigator is now able to capture or modify the traffic. The successful *MitM* attack is established.

enclosed harmful content. *DNS Spoofing* can be effectively applied for spoofing fake websites (Prowell et al., 2010, p. 112). If the attacker detects a *DNS* message, he intercepts it and forges a reply for the victim. The victim receives forged mapping of domain name to *IP* address and starts communication with the fake device without noticing the attack. The attacker then acts as the inquired service and therefore performs a *MitM* attack.

2.4 Available Tools for Specific Phases of the MitM Attack on Wireless Networks

From perspective of the intended functionality of the implemented tool, the whole process of *MitM* attack on wireless networks can be divided into three main phases: *Accessing wireless network*, *Tampering network topology* and *Capturing network traffic*, as explained in Figure 3.

To access secured wireless networks, *Aircrack-ng suite*[7] is considered a reliable software solution. Considering the phase *Accessing wireless network* (Figure 3), following tools were utilized. *Airmon-ng* can manage modes of a wireless interface. *Airodump-ng* can be used to scan and detect attacked *AP*. *Aircrack-ng* together with *aireplay-ng*, *airodump-ng* and *upc_keys* can be utilized for cracking *WEP OSA*, *WEP SKA*, *WPA PSK* and *WPA2 PSK*. The tool *wifiphisher*[8] can be used to perform impersonation and phishing. Connection to the wireless network can be established by *netctl*[9].

MITMf[10] with its *Spoof* plugin can be used during the *Tampering network topology* phase. For the realization of *DNS Spoofing*, it is possible to use tool *dnsspoof*, which is a part of *dsniff* collection (Song, 2001). This collection of network auditing and penetration testing tools contains several advanced programs, which could be used for tampering network topology.

Capturing traffic can be done by the tool *dumpcap*[11], which is part of the *Wireshark*[12] distribution. Behaviour, usage and success rate of individual tools, as well as possibilities of controlling them by the implemented tool, were analyzed. The software selected for individual tasks of the automated *MitM* attack were chosen from the researched variety of available tools based on performed manual experiments, further described in the thesis (Vondráček, 2016).

3. ATTACK AUTOMATION USING WIFIMITM PACKAGE AND WIFIMITMCLI TOOL

The implemented tool is currently intended to run on *Arch Linux*[13], but it could be used on other platforms which would satisfy specified dependencies. This distribution was selected because it is very flexible and lightweight. Python 3.5 was selected as a primary implementation language for the automated tool and Bash was chosen for supporting tasks, e.g., installation of dependencies on *Arch Linux* and software wrappers.

The functionality implemented in the *wifimitm* package could be directly incorporated into other software products based on Python language. This way

[7] http://www.aircrack-ng.org/

[8] https://github.com/sophron/wifiphisher

[9] https://www.archlinux.org/packages/core/any/netctl/

[10] https://github.com/byt3bl33d3r/MITMf

[11] https://www.wireshark.org/docs/man-pages/dumpcap.html

[12] https://www.wireshark.org/

[13] https://www.archlinux.org/

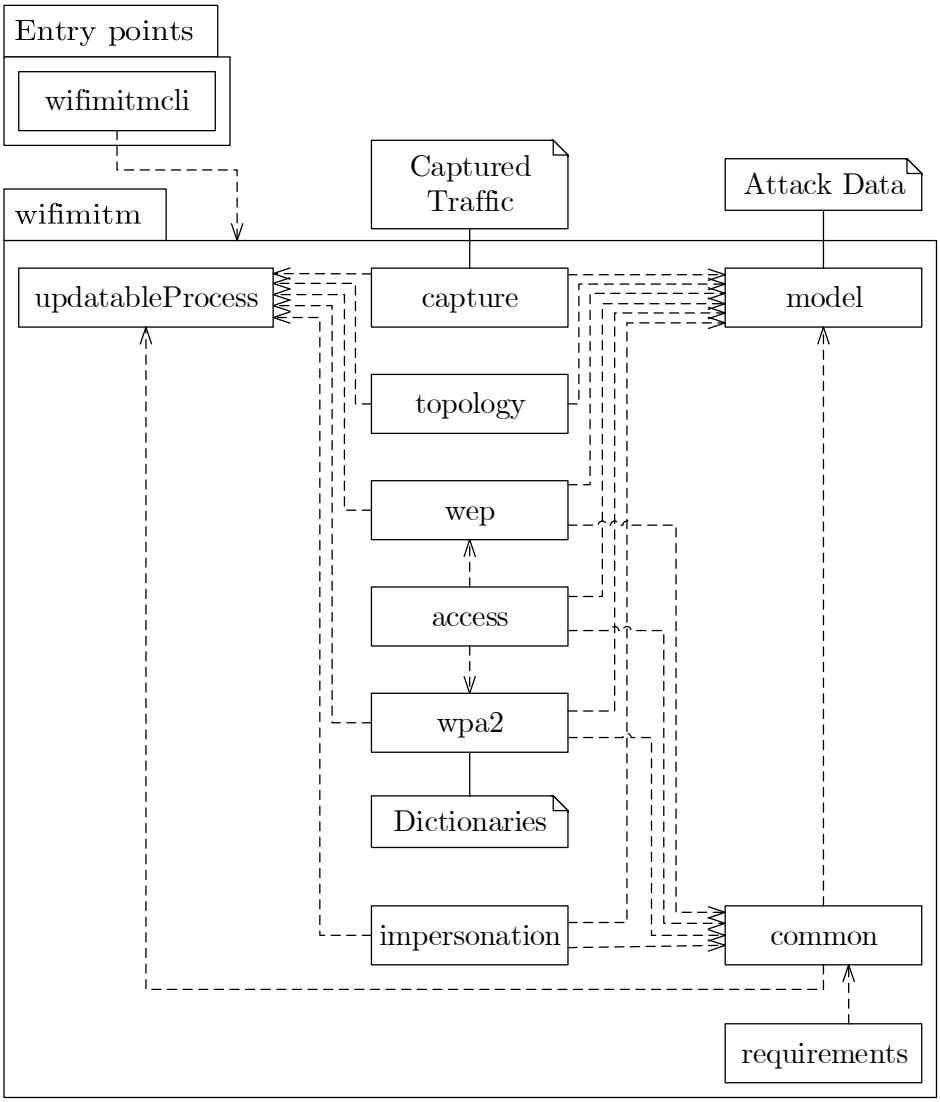

Figure 4. This figure shows the basic structure of the developed application. The tool *wifimitmcli* uses a functionality offered by the package *wifimitm*. The package is also able to manipulate attack data useful for repeated attacks and capture files with intercepted traffic. Detailed structure of the package is described in section 3.

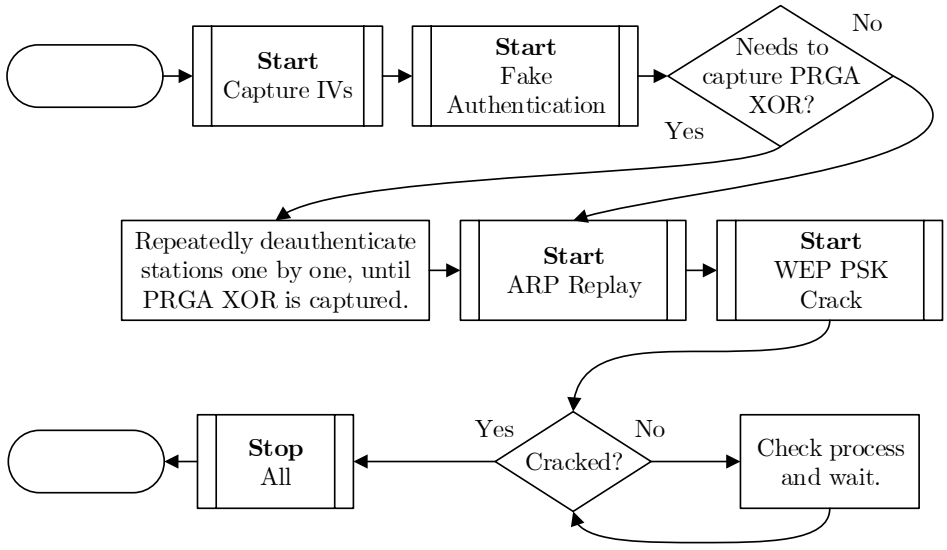

Figure 5. The figure shows a simplified flowchart of cracking *WEP OSA* or *WEP SKA* secured wireless network. Cracking procedure is a part of the first phase *Accessing wireless network* as described in Figure 3. If the given *WLAN* has already been successfully attacked, *Attack Data* (Section 3.1) contains the correct key. In such cases, repetitive cracking is unnecessary and is therefore skipped.

the package would work as a software library. Schema of the *wifimitm* package is in Figure 4.

The *wifimitm* package consists of following modules. The `access` module offers an automated process of cracking selected *WLAN*. It uses modules `wep` and `wpa2`, which implement attacks and cracking based on the used security algorithm. The `wep` module is capable of fake authentication with the *AP*, *ARP replay* attack (to speed up gathering of *IV*s) and cracking the key based on *IV*s. In the case of *WPA2* secured network, the `wpa2` module can perform a dictionary attack, personalize used dictionary and verify a password obtained by phishing (Figure 4). Verification of the password and dictionary attacks are done with a previously captured handshake. The `common` module contains functionality which could be used in various parts of the process for scanning and capturing wireless communication in monitor mode. The `common` module also offers a way to deauthenticate *STA*s from selected *AP*.

If a dictionary attack against a correctly secured network fails, a phishing attack can be managed by the `impersonation`[14] module. The `topology` module can be used to change network topology. It provides functionality for *ARP Spoofing*. The `capture` module focuses on capturing network traffic (Figure 4). It is intended to be used after the tool is successfully connected to the attacked network and network topology was successfully changed into the one suitable for *MitM* attack.

[14]For details concerning individual phishing scenarios, please see *wifiphisher*'s website. `https://github.com/sophron/wifiphisher`

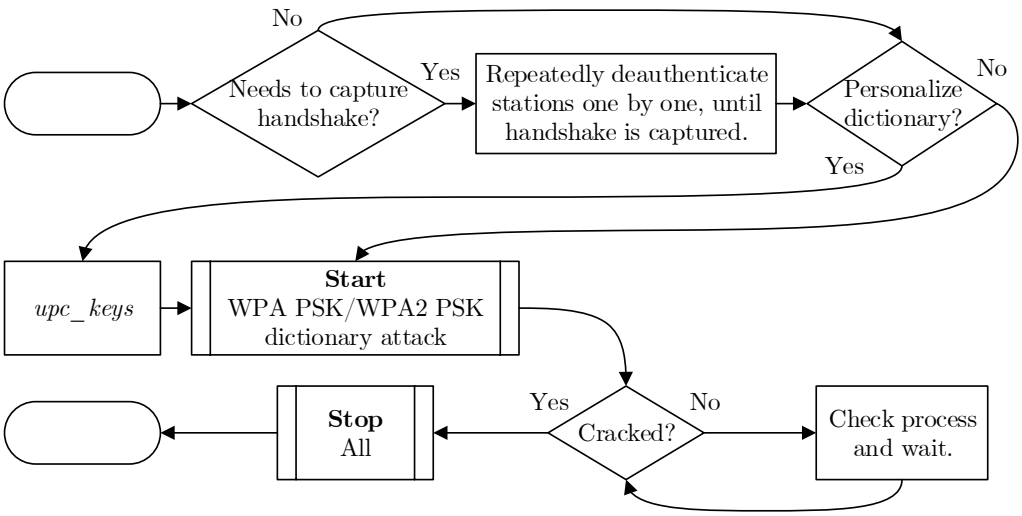

Figure 6. This simplified flowchart illustrates cracking *WPA PSK* or *WPA2 PSK* secured network. Similarly as cracking in Figure 5, this procedure is also a part of the first phase *Accessing wireless network* (Figure 3). Cracking can also be skipped if the key is already known. As already described, impersonation (phishing) can be used in a case of unsuccessful cracking.

3.1 Attack Data

Various attacks executed against the selected *AP* require some information to be captured first. ARP request replay attack on *WEP* secured networks requires an ARP request to be obtained in order to start an attacking procedure. Fake authentication in *WEP SKA* secured network needs *PRGA XOR*[15] obtained from a detected authentication. Dictionary attack against *WPA PSK* and *WPA2 PSK* secured networks requires a captured handshake. Finally, for the successful connection to a network, a correct key is required. When the required information is obtained, it can be saved for a later usage to speed up following or repetitive attacks. Data from successful attacks could be even shared between users of the implemented tool.

[15]Stream of *Pseudo Random Generation Algorithm* generated bits.

3.2 Dictionary Personalization

Weaknesses in default network passwords could be exploited to improve dictionary attacks against *WPA PSK* and *WPA2 PSK* security algorithms. The implemented tool incorporates *upc_keys* for generation of possible default passwords if the selected network matches the criteria. The *upc_keys* tool generates passwords, which are transferred to the cracking tool using pipes. With this approach, the implemented tool could be further improved for example to support localized dictionaries.

3.3 Requirements

The implemented automated tool depends on several other tools, which are being controlled. The Python package can be automatically installed by its setup including Python dependencies. Non-Python dependencies can be satisfied by installation

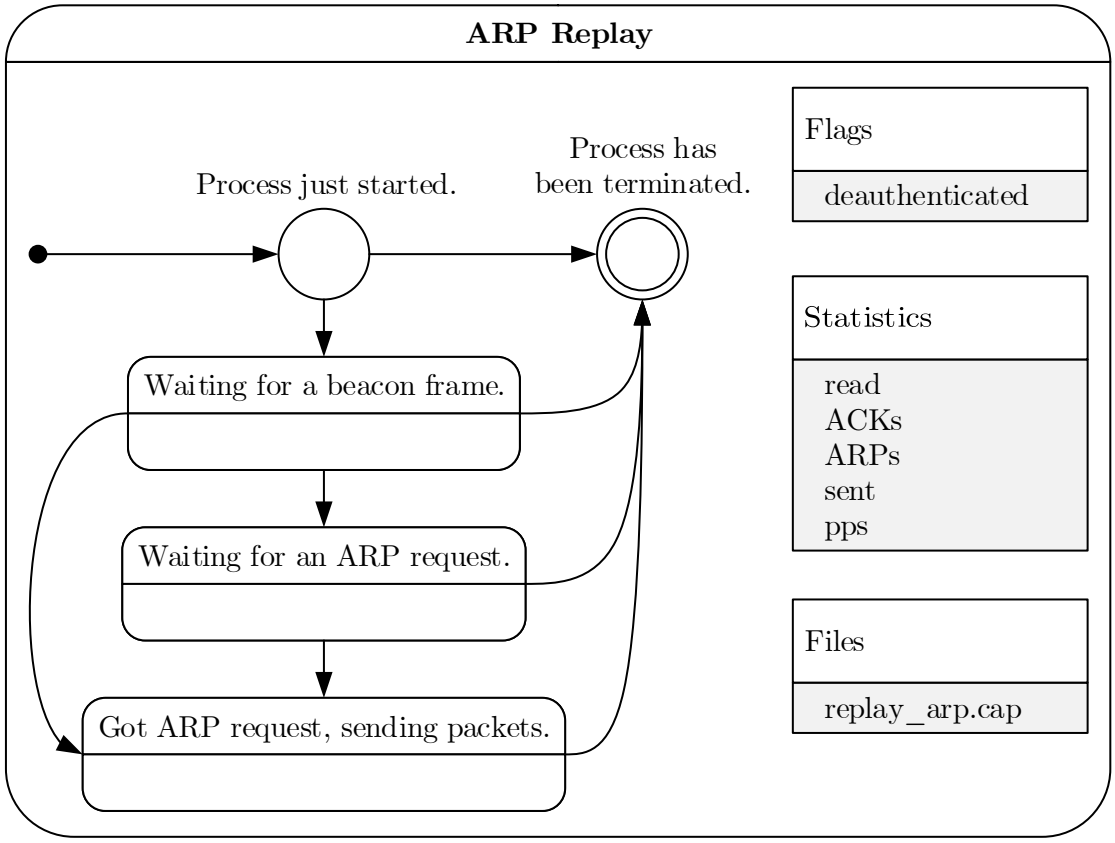

Figure 7. This figure presents the information model of a process controlled by *wifimitm*. In this example, the incorporated tool is *aireplay-ng* from *Aircrack-ng suite* executing *ARP replay* attack to speed up gathering of *IV*s. State of the process is modeled using a *FSM* consisting of 5 states. In a case that the attacking device receives at least one deauthentication packet, the deauthenticated flag is set. Statistics contain overall information about processed packets. Useful file created by *aireplay-ng* during this procedure is a capture file containing *ARP* request. This file is part of the *Attack Data*, as outlined in Section 3.1.

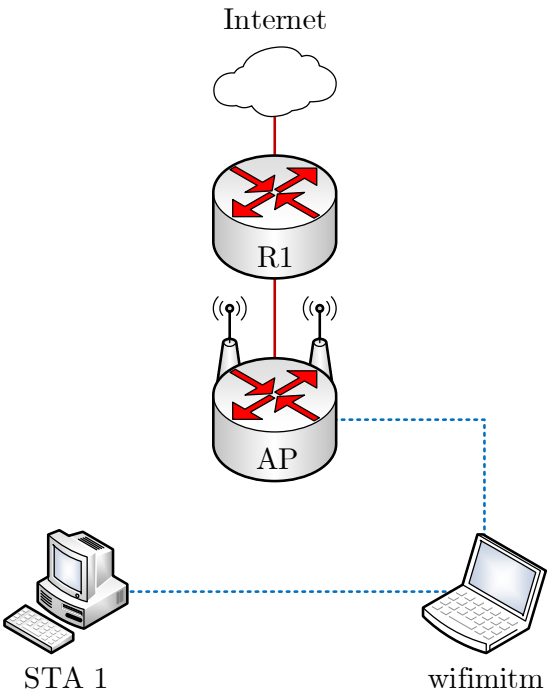

Figure 8. This figure presents the network topology used for the first performance testing (Section 4.1) and success rate measurements (Section 4.2). Results of this performance testing are in Figure 10.

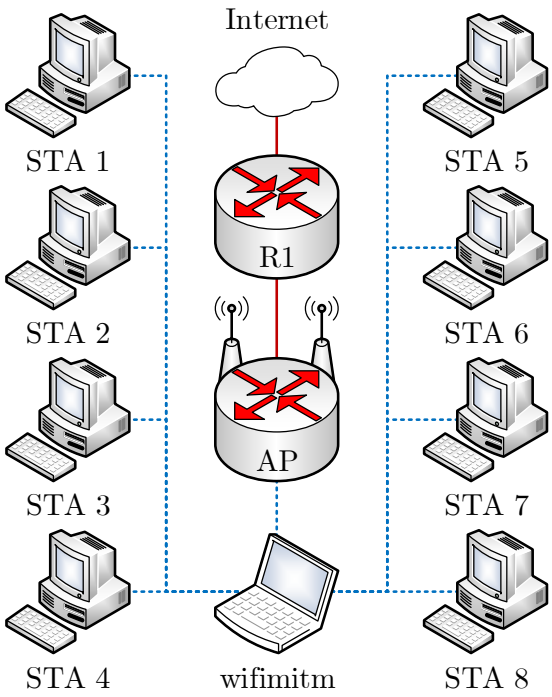

Figure 9. This figure shows the network topology consisting of 8 *STA*s and 1 *AP* which was used for the second performance testing (Section 4.1). Results of this performance testing are in Figure 11.

scripts and wrappers, which are currently developed for *Arch Linux*.

MITMf has a number of dependencies. Therefore, the installation script also creates a virtual environment dedicated to *MITMf*. After installation, *MITMf* can be easily run encapsulated in its environment. *Wifiphisher* is also installed in a virtualized environment and run using a wrapper. Tool *upc_keys* is compiled during installation. Some changes in *wifiphisher*'s source code were implemented, the installation script therefore applies a software patch. Other software dependencies are installed using a package manager.

Due to the nature of concrete steps of the attack, a special hardware equipment is required. During the scanning and capturing of network traffic without being connected to the network, an attacking device needs a wireless network interface in *monitor mode*. For sending forged packets, the wireless network interface also needs to be capable of *packet injection*. To be able to perform a phishing attack, a second wireless interface capable of master (*AP*) mode has to be available. The user can check whether his hardware is capable of packet injection using the *aireplay-ng* tool. Managing monitor mode of interface is possible with the *airmon-ng* tool.

3.4 Incorporation of tools

The implemented tool needs to interact with other software tools in order to automate attack procedures. Incorporated tools com-

municate using *Standard output stream* (*stdout*), *Standard error stream* (*stderr*) and optionally using generated files. *Wifimitm* needs to continuously analyze all these outputs to be aware of current state of the controlled tool. Information contained in the output can be a summary of current progress, a notification that some event occurred or a result of an intended action.

To meet requirements for efficient incorporation of other tools so that the *wifimitm* package could interact with them, the *updatableProcess* module was developed. This module contains an abstract base class *UpdatableProcess*. Individual incorporated tools have dedicated classes inherited from the *UpdatableProcess* which are used for managing these tools from *wifimitm*. When a process is spawned, using an instance of class inherited from *UpdatableProcess*, it is assigned a temporary directory for its outputs. The running process is continuously writing to *stdout* and *stderr*. The outputs are periodically analyzed. Classes inherited from *UpdatableProcess* can implement a signalization of process' state using a *Finite State Machine* (*FSM*). Process' output can include notifications of events. Upon detection of such event, appropriate flags can be set. Some processes also output summary information, which can be used to update statistics. Continuously updated information about the process can therefore consist of state, flags, statistics and created files as presented in Figure 7.

4. EVALUATION

The capabilities of the implemented tool were evaluated. Because the tool deploys man-in-the-middle type of attack, the tool necessary modifies the target environment. Thus we evaluated the footprint of the tool and the possibility to detect the running attack by the victim. The next set of experiments were conducted to show how easy it is to gain the network communication for different wireless configurations.

4.1 Attack's Performance Impact

The first experiment examines wheathe the attack is observable from end-user perspective or disrupts regular communication on the network. A scheme of the networks used for this experiment is shown in Figures 8 and 9, modeling SOHO[16] environment. The *STAs* were correctly connected to the *AP*, and they were successfully communicating with the Internet. The implemented *wifimitmcli* tool was then started and automatically attacked the network, as described in Section 3 and Figure 3.

The performance impact of the *wifimitm* was compared using typologies presented in Figures 8 and 9. As the observed metric was selected a *Round-Trip Time* (*RTT*) value describing a delay that end-user might experience when the load on the *R1* is increased.

For the first case, only one client is connected at the time. The Figure 10 plots *RTT* values measured between *STA1* and its Internet gateway *R1*. The x axe denotes each measurement, and on the y axe is shown corresponding delay in ms in a logarithmic scale.

The second case shows eight *STAs* connected to *R1* simultaneously, in Figure 9. Figure 11 shows an increase of *RTT* measured between each of *STAx* and *R1*.

Both cases were evaluated on the fact, whether the attack being performed was revealed or whether the users had any suspicion about the malicious transformation of their *WLAN*. By results comparison of both test cases, presented in Figures 10 and 11, can be concluded that regular user has no way of knowing whether the increase of la-

[16] small office/home office

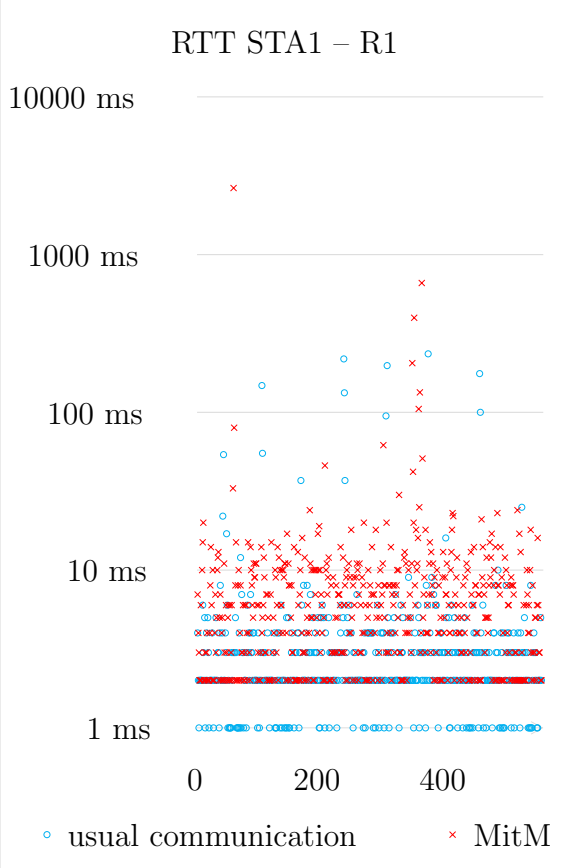

Figure 10. The first *WLAN* for performance testing was the same as for the success rate measurements described in Section 4.2. Figure shows comparison of the measured *RTT* between *STA1* and *R1* during usual communication and during successful *MitM* attack. The results show the performance impact is not critical. Discussion with the users of the attacked network proved this attack unrecognizable.

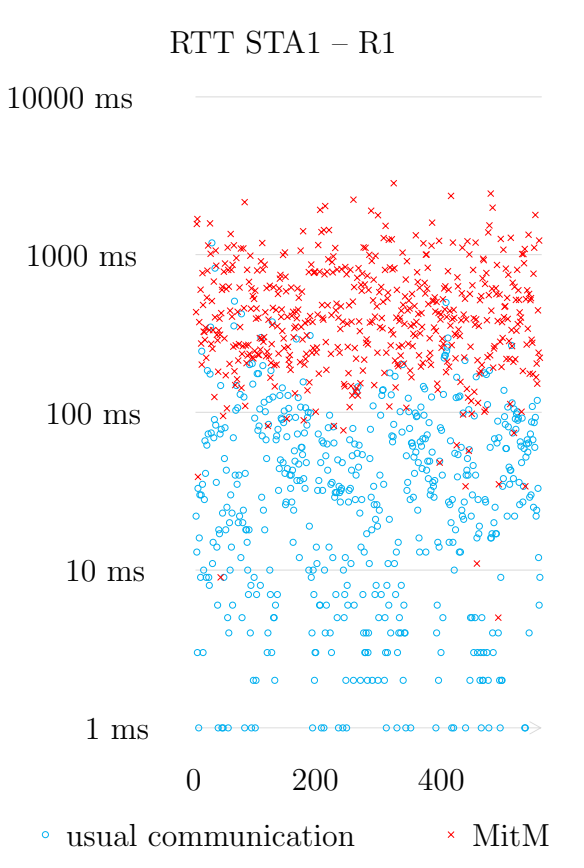

Figure 11. The second performance testing consisted of 8 *STA*s and 1 *AP* connected to the Internet – streaming videos, downloading large files, etc. The figure compares the *RTT* between *STA1* and *R1* similarly. The performance impact is more severe than in Figure 10. Despite the performance impact, the users had no suspicion that they were under *MitM* attack. Instead, they blamed the amount of devices for network congestion.

tency is caused by an attack, or by a new device connecting to the network, massive data transfer, or any other interference from the physical world.

On the other hand, there is apparent linear segregation between measurements with and without attack in Figure 11. This observation submits a future challenge, whether this condition might be used as a feature for a wireless network diagnostic without direct access to *R1* or any of *STAs*.

4.2 Experiments Concerning Various Network Devices and Configurations

The second experiment observes applicability of the *wifimitmcli* tool in different SOHO environments based on multiple AP devices with a variety of commonly used security settings in combination with numerous end-user devices. The experiment was considered successful if the *wifimitmcli* was able to perform all phases of MitM attack, Figure 3, and place itself in the middle of communication to capture network traffic according to the concept of *MitM*, Section 2.3 and Figure 2. For the test case to be correct, no help from the investigator was allowed during the attack performed by *wifimitmcli*.

The first use-case was to test all combinations of available AP devices with all available client ones. Figure 8 shows network topology used in this controlled laboratory experiment. Results of the success rate measurements are shown in Tables 3 and 4.

The second use-case was to test success rate of the *wifimitmcli* tool in a non-laboratory environment beyond our control on the end-user part. Figure 8 shows once again testing topology with *Linksys WRP400* device as an AP. Table 4 shows measurements and success rate of observations of this use-case. The experiment was conducted during the author's presentation at the Brno University of Technology, Faculty of Information Technology where visitors were invited to let their devices be attacked.

Results of experiments present in Tables 3, 4 and the thesis (Vondráček, 2016, pp. 42–43) reveal the following conclusions:

- Open – networks can be very easily attacked.
- *WEP OSA* and *WEP SKA* – secured networks can be successfully attacked even if they use a random password.
- *WPA PSK* and *WPA2 PSK* – secured networks suffer from weak passwords (dictionary attack), default passwords and mistakes of users (impersonation and phishing).

Consequently, results reveal feasibility and ease of *MitM* attack using the *wifimitm*, and its success rate in the target SOHO environments.

5. CONCLUSIONS

The goal of this research was to implement a tool that would be able to automate all the necessary steps to perform *MitM* attacks on *WLAN*s. The authors searched for and analyzed a range of software and methods focused on penetration testing, communication sniffing and spoofing, password cracking and hacking in general. To be able to design, implement and test the tool capable of such attacks, knowledge of different widespread security approaches was essential. The authors further focused on possibilities of *MitM* attacks even in cases where the target *WLAN* is secured correctly. Therefore, methods and tools for impersonation and phishing were also analyzed.

The authors' work and research resulted in creation of the *wifimitm* Python package.

Table 3. This table presents results of the success rate measurements. A successful attack is marked using a *checkmark* symbol (✓) and unsuccessful attack is marked using a *times* symbol (×). In the case when the attack was not fully successful, the question mark (?) is used. Such partially successful test (? symbol) can for example happen in situation where the suspect is sending only a portion of his traffic through the investigator. Some of the used *STA*s lack *WEP SKA* settings (☐ symbol). Testing *WPA PSK* and *WPA2 PSK* networks were configured with password "`12345678`" and *WEP* secured networks used password "`A_b#1`".

		Lenovo G580, Windows 10	Lenovo G505s, Windows 8.1	Dell Latitude E6500, Ubuntu 17.04	HTC Desire 500, Android 4.1.2	Apple iPhone 4, iOS 7.1.2
Linksys WRT610N	*open*	✓	✓	✓	✓	✓
	WEP OSA	✓	✓	✓	✓	✓
	WEP SKA	☐	☐	✓	✓	✓
	WPA PSK	✓	✓	✓	✓	✓
	WPA2 PSK	✓	✓	✓	✓	✓
Linksys WRT54G	*open*	✓	✓	✓	✓	✓
	WEP OSA	✓	✓	✓	✓	✓
	WEP SKA	☐	☐	✓	✓	✓
	WPA PSK	✓	✓	✓	✓	✓
	WPA2 PSK	✓	✓	✓	✓	✓
Linksys WRP400	*open*	✓	✓	✓	✓	✓
	WEP OSA	✓	✓	✓	✓	✓
	WEP SKA	☐	☐	✓	✓	✓
	WPA PSK	✓	✓	✓	✓	✓
	WPA2 PSK	✓	✓	✓	✓	✓
TP-LINK TL-WR841N	*open*	?	×	✓	✓	✓
	WEP OSA	?	×	✓	✓	×
	WEP SKA	☐	☐	✓	✓	×
	WPA PSK	?	×	✓	✓	×
	WPA2 PSK	?	×	✓	✓	×
D-Link `DVA-G3671B`	*open*	✓	✓	✓	✓	✓
	WEP OSA	✓	✓	✓	✓	✓
	WEP SKA	☐	☐	✓	✓	✓
	WPA PSK	✓	✓	✓	✓	✓
	WPA2 PSK	✓	✓	✓	✓	✓

Table 4. The following table shows the results of public experiments. Testing network utilized *Linksys WRP400* as an AP and end-user devices of random people that agreed to participate in the experiment. A successful attack is marked using a *checkmark* symbol (✓).

Model	OS	Attack
HTC Desire 500	Android 4.1.2	✓
HTC Desire 820	Android 6.0.1	✓
Apple iPhone 6	iOS 10.3.1	✓
Apple iPhone 5s	iOS 10.2.1	✓
Apple iPhone 5	iOS 10.3.1	✓
Apple iPhone 5c	iOS 9.2.1	✓
Apple iPhone 4	iOS 7.1.2	✓

This package serves as a library which provides functionality for automation of *MitM* attacks on target *WLAN*s. The developed package can also be easily incorporated into other tools. Another product of this research is the *wifimitmcli* tool which incorporates the functionality of the *wifimitm* package. This tool automates the individual steps of a *MitM* attack and can be used from a *CLI*. The implemented software comes with a range of additions for convenient usage, e.g., a script that checks and installs dependencies on *Arch Linux*, a Python setuptools setup script and of course a manual page.

The *wifimitmcli* tool, and therefore *wifimitm* as well, was tested during experiments with an available set of equipment. As the results show, the implemented software product is able to perform an automated *MitM* attack on *WLAN*s successfully.

Upon successful deployment and execution of the implemented tool, an investigator can eavesdrop or spoof the passing communication. The goal of the tool was to automate *MitM* attacks on *PSK* secured *WLANs*. It does not focus on dissecting further traffic protections. This means that it does not interfere with *SSL/TLS*, *VPN*, or other encapsulations. Thanks to the tool's design, it can be easily used together with other software specialized on interception of encapsulated traffic. Traffic encapsulation is a sufficient protection against this tool. From the *WLAN* administrators point of view, available defense mechanisms are outlined in Section 2.2.

As explained earlier, all the suspect's network traffic is passing through the attacking device during a successful *MitM* attack. Unfortunately, there could be users on the network other than the ones that are subject to a court order. Making sure that only appropriate traffic is being captured may be important depending on the nature of the court order or the legislation. This challenge may be solved by setting corresponding filter rules for traffic capture software.

This research and its products can be utilized in combination with other security research carried out at the Brno University of Technology, Faculty of Information Technology. It can serve in investigations done by forensic researchers (Pluskal et al., 2015). It can also be used in automated penetration testing of *WLANs*.

In the future iterations of the development, the product could focus on exploiting the weaknesses of the widely used *WPS* technology, incorporating techniques to perform *KRACKs*, or focus on detection of attacks themselves. Concerning the current state of the product, it does not focus on enterprise *WLAN*s, which also suffer from their weaknesses.

ACKNOWLEDGEMENTS

This work was supported by Ministry of Interior of the Czech Republic project "Integrated platform for analysis of digital data

from security incidents" VI20172020062; Ministry of Education, Youth and Sports of the Czech Republic from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science" LQ1602; and by BUT internal project "ICT tools, methods and technologies for smart cities" FIT-S-17-3964.

AUTHOR BIOGRAPHIES

Martin Vondráček is currently pursuing Master's degree in Intelligent Systems at the Brno University of Technology. He has completed the Bachelor's degree with honours at the same university in 2016 and received dean's award and rector's award later that year. His interest in Computer Science increased during his thesis work at the University of Malta in 2016 and exchange study of Computer Forensics at the University of South Wales in 2017. He presented outcomes of his research at conferences ICDF2C and Excel@FIT. He is dedicated to research, information security, computer networks and software development.

Jan Pluskal is a Ph.D. student at FIT BUT, freelance lecturer and programmer. He is mostly interested in computer network forensics, machine learning, distributed computing and C# programming. The main author of Netfox Detective – a tool for network forensic analysis. In his free time, he plays around with home automation IoT technologies to ease up daily routines of his family.

Ondřej Ryšavý received the Ph.D. degree in computer science from the Brno University of Technology, Brno, Czech Republic, in 2005. He is an Associate Professor with the Department of Information Systems, Brno University of Technology, Brno. His research interests include computer networking and, in particular, network monitoring, network security and forensics, and network architectures. His work is focused on improving network security through data analysis by application of data mining, statistics, and distributed computing.

REFERENCES

Callegati, F., Cerroni, W., & Ramilli, M. (2009, Jan). Man-in-the-middle attack to the HTTPS protocol. *Security Privacy, IEEE*, 78–81. doi: 10.1109/MSP.2009.12

Cisco Systems, Inc. (2013). *Catalyst 6500 release 12.2sx software configuration guide*. Retrieved on January 29, 2018, from `http://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst6500/ios/12-2SX/configuration/guide/book.html`

Deal, R., & Cisco Systems, I. (2006). *The complete cisco vpn configuration guide*. Cisco Press. Retrieved on January 30, 2018, from `https://books.google.cz/books?id=ms-8AAAACAAJ`

Droms, R. (1997, March). *Dynamic Host Configuration Protocol* (RFC No. 2131). Internet Engineering Task Force. RFC 2131 (DRAFT STANDARD). Retrieved on January 30, 2018, from `http://www.ietf.org/rfc/rfc2131.txt`

Fluhrer, S., Mantin, I., & Shamir, A. (2001). Weaknesses in the key scheduling algorithm of RC4. In S. Vaudenay & A. Youssef (Eds.), *Selected areas in cryptography* (pp. 1–24). Springer Berlin Heidelberg. Retrieved on January 30, 2018, from `http://dx.doi.org/10.1007/3-540-45537-X_1` doi: 10.1007/3-540-45537-X_1

Godber, A., & Dasgupta, P. (2003). Countering rogues in wireless

networks. In *Proceedings of the international conference on parallel processing workshops* (Vol. 2003-January, pp. 425–431). Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/ICPPW.2003.1240398

Halsall, F. (2005). *Computer networking and the internet*. Addison-Wesley. Retrieved on January 22, 2016, from https://books.google.cz/books?id=QadX5XErZ9IC

Heffner, C. (2011). *Cracking WPA in 10 hours or less – /dev/ttys0.* Retrieved on April 4, 2016, from http://www.devttys0.com/2011/12/cracking-wpa-in-10-hours-or-less/

IEEE-SA. (2012, March). IEEE standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, 1–2793. doi: 10.1109/IEEESTD.2012.6178212

Kent, S., & Seo, K. (2005, December). *Security Architecture for the Internet Protocol* (RFC No. 4301). Internet Engineering Task Force. RFC 4301 (PROPOSED STANDARD). Retrieved on January 30, 2018, from https://www.ietf.org/rfc/rfc4301.txt

Klinec, D., & Svítok, M. (2016a). *UPC UBEE EVW3226 WPA2 password reverse engineering, rev 3.* Retrieved on January 30, 2018, from https://deadcode.me/blog/2016/07/01/UPC-UBEE-EVW3226-WPA2-Reversing.html

Klinec, D., & Svítok, M. (2016b). *Wardriving Bratislava 10/2016.* Retrieved on January 30, 2018, from https://deadcode.me/blog/2016/11/05/Wardriving-Bratislava-10-2016.html

Kumkar, V., Tiwari, A., Tiwari, P., Gupta, A., & Shrawne, S. (2012). Vulnerabilities of wireless security protocols (WEP and WPA2). *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, *1*(2), 34–38. Retrieved on January 30, 2018, from http://ijarcet.org/wp-content/uploads/IJARCET-VOL-1-ISSUE-2-34-38.pdf

Liu, Y., Jin, Z., & Wang, Y. (2010, Sept). Survey on security scheme and attacking methods of WPA/WPA2. In *2010 6th international conference on wireless communications networking and mobile computing (wicom)* (pp. 1–4). doi: 10.1109/WICOM.2010.5601275

Plummer, D. (1982, November). *Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware* (RFC No. 826). Internet Engineering Task Force. RFC 826 (INTERNET STANDARD). Retrieved on January 30, 2018, from http://www.ietf.org/rfc/rfc826.txt

Pluskal, J., Matoušek, P., Ryšavý, O., Kmeť, M., Veselý, V., Karpíšek, F., & Vymlátil, M. (2015). Netfox detective: A tool for advanced network forensics analysis. In *Proceedings of security and protection of information (spi) 2015* (pp. 147–163). Brno University of Defence.

Retrieved on January 30, 2018, from http://www.fit.vutbr.cz/research/view_pub.php?id=10863

Prowell, S., Kraus, R., & Borkin, M. (2010). Chapter 6 - man-in-the-middle. In S. Prowell, R. Kraus, & M. Borkin (Eds.), *Seven deadliest network attacks* (pp. 101–120). Boston: Syngress. Retrieved on January 30, 2018, from http://www.sciencedirect.com/science/article/pii/B9781597495493000067 doi: http://dx.doi.org/10.1016/B978-1-59749-549-3.00006-7

Robyns, P. (2014). *Wireless network privacy* (Master's thesis, Hasselt University, Hasselt). Retrieved on January 30, 2018, from http://hdl.handle.net/1942/17516

Song, D. (2001, Dec). *dsniff*. Retrieved on January 27, 2018, from http://www.monkey.org/~dugsong/dsniff

Tews, E., Weinmann, R.-P., & Pyshkin, A. (2007). Breaking 104 bit WEP in less than 60 seconds. In S. Kim, M. Yung, & H.-W. Lee (Eds.), *Information security applications* (pp. 188–202). Springer Berlin Heidelberg. Retrieved on January 30, 2018, from http://dx.doi.org/10.1007/978-3-540-77535-5_14 doi: 10.1007/978-3-540-77535-5_14

Thomas, O. (2017). *Windows server 2016 inside out*. Pearson Education. Retrieved on January 30, 2018, from https://books.google.cz/books?id=rLfDDgAAQBAJ

Vanhoef, M., & Piessens, F. (2017). Key reinstallation attacks: Forcing nonce reuse in WPA2. In *Proceedings of the 24th acm conference on computer and communications security (ccs)*. ACM.

Vondráček, M. (2016). *Automation of MitM attack on WiFi networks* (Bachelor's thesis, Brno University of Technology, Faculty of Information Technology). Retrieved on January 30, 2018, from http://www.fit.vutbr.cz/study/DP/BP.php?id=18596

Vondráček, M., Pluskal, J., & Ryšavý, O. (2018). Automation of MitM attack on Wi-Fi networks. In P. Matoušek & M. Schmiedecker (Eds.), *Digital forensics and cyber crime* (pp. 207–220). Cham: Springer International Publishing.

