Automation of MitM Attack on Wi-Fi Networks

Martin Vondráček(✉), Jan Pluskal, and Ondřej Ryšavý

Brno University of Technology, Božetěchova 2, Brno, Czech Republic
xvondr20@stud.fit.vutbr.cz, {ipluskal,rysavy}@fit.vutbr.cz
http://www.fit.vutbr.cz/
https://mvondracek.github.io/wifimitm/

Abstract. Security mechanisms of wireless technologies often suffer weaknesses that can be exploited to perform Man-in-the-Middle attacks, allowing to eavesdrop or to spoof network communication. This paper focuses on possibilities of automation of these types of attacks using already available tools for specific tasks. Outputs of this research are the *wifimitm* Python package and the *wifimitmcli CLI* tool, both implemented in Python. The package provides functionality for automation of *MitM* attacks and can be used by other software. The *wifimitmcli* tool is an example of such software that can automatically perform multiple *MitM* attack scenarios without any intervention from an investigator.

The results of this research are intended to be used for automated penetration testing and to help with forensic investigation. Finally, a popularization of the fact that such severe attacks can be easily automated can be used to raise public awareness about information security.

Keywords: Man-in-the-Middle attack
Accessing secured wireless networks · Password cracking
Dictionary personalization · Tampering network topology
Impersonation · Phishing

1 Introduction

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 technologies 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 [1], [16, pp. 101–120].

In an example scenario, 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 ISP[1] and harden his network.

[1] Internet Service Provider

P. Matoušek and M. Schmiedecker (Eds.): ICDF2C 2017, LNICST 216, pp. 207–220, 2018.
https://doi.org/10.1007/978-3-319-73697-6_16

For example, he could use an overlay network technology, e.g., *VPN* (implemented by *L2TP*, *IPsec* [9, 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 [5, pp. 425–431], but this scheme, or similar, is often used by large vendors like Cisco [2] or Microsoft [19] for branch office deployment and can also be seen in home routers[2]. 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.

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 Fig. 1, capable of accessing and acquiring evidence from a wireless network regardless of used security mechanisms. This framework can be split into several steps. First, 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. 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 Sect. 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*. 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 final implementation on carefully devised experiments,

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

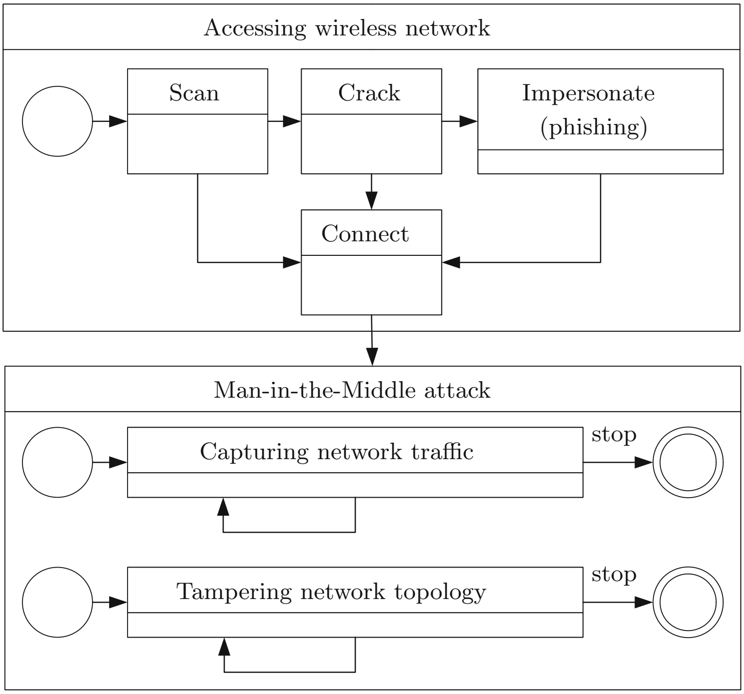

Fig. 1. During the first phase – *Accessing wireless network*, the tool is capable of an attack on *WEP OSA*, *WEP SKA*, *WPA PSK* and *WPA2 PSK* secured *WLAN*s. In a case of the dictionary attack on the device deployed by the UPC company, used dictionaries are personalized by the implicit passwords. 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.

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 (Sects. 2.1 and 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 [6, p. 665], [8, pp. 1167–1169]. At this point, *WEP* is deprecated and superseded by subsequent algorithms, but is still sometimes used, as can be seen from Table 1 available from *Wifileaks.cz*[3]. *WEP* suffers from weaknesses and, therefore, it has been broken [4]. There are already implemented tools to provide access to wireless networks secured by *WEP* available [18]. Regarding *WEP* secured *WLAN*s, authentication can be either *Open System Authentication* (*OSA*) or *Shared Key Authentication* (*SKA*) [8, pp. 1170–1174]. In the case of *WEP OSA*, any *station* (*STA*) can successfully authenticate to the *Access Point* (*AP*) [17, 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).

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%

Wi-Fi Protected Access® (*WPA*) was developed by the Wi-Fi Alliance® as a reaction to increasing number of security flaws in *WEP*. The main flaw of *WPA* security algorithm can be identified at the beginning of client device's communication, where an unsecured exchange of confidential information is performed during the four-way handshake. An investigator can obtain this unsecured communication and use it for consecutive cracking of the *Pre-Shared Key* (*PSK*).

Wi-Fi Protected Access® *2* (*WPA2*™) is a successor of *WPA*, but security flaws of the *WPA PSK* algorithm remain significant also for the *WPA2 PSK*. Information exposed during the handshake can be used for the dictionary attack, which can be further improved by precomputing the *Pairwise Master Keys* (*PMKs*) [12, pp. 37–38], [13, p. 3]. Precomputed lookup tables are already available online[4].

A critical security flaw in wireless networks secured by *WPA* or *WPA2* is the functionality called *Wi-Fi Protected Setup*™ (*WPS*). This technology was introduced with an aim to provide a comfortable and secure way of connecting

[3] http://www.wifileaks.cz/statistika/
[4] https://www.renderlab.net/projects/WPA-tables/

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 [7]. 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*[5].

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. In a case of possible 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*[6] by Peter Geissler.[7] 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.

Table 2. Results of wardriving in Bratislava and Brno focused on UPC vulnerabilities concerning default *WPA2 PSK* passwords [11]. Detailed article about these security flaws is available online [10].

Bratislava (capital of Slovakia) 2016-10-01	Count	Ratio
Total networks	22 172	
UPC networks	3 092	13.95%
UPC networks, vulnerable	1 327	42.92% UPC
Brno (city in the Czech Republic) 2016-02-10	Count	Ratio
Total networks	17 516	
UPC networks	2 868	16.37%
UPC networks, vulnerable	1 835	63.98% UPC

2.2 Network Technologies Used in WLANs

In the context of a MitM attack on a *WLAN*, we are targeting some common network protocols:

– *DHCP* automates network device configuration without a user's intervention [3].

[5] https://code.google.com/archive/p/reaver-wps/
[6] https://haxx.in/upc-wifi/
[7] UPC company is a major ISP in the Czech Republic, URL: https://www.upc.cz

- *ARP* translates an *IPv4* address to a destination *MAC* address of the next-hop device in the local area network [14].
- *IPv6* networks utilize *ICMPv6 Neighbor Discovery* functionality to achieve similar functionality to *ARP* in *IPv4* networks.

These network protocols are vulnerable and a *MitM* attack is a coordinated attack on each of these protocols, effectively changing the network topology.

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

ARP Spoofing technique was selected from the researched methods. This method proved itself with reasonable performance during experiments. Possible countermeasures to these attacks are further described in the thesis [20].

2.3 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 Fig. 1.

To access secured wireless networks, *Aircrack-ng suite*[8] is considered a reliable software solution. Considering the phase *Accessing wireless network* (Fig. 1), 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*[9] can be used to perform impersonation and phishing. Connection to the wireless network can be established by *netctl*[10]. *MITMf*[11] with its *Spoof* plugin can be used during the *Tampering network topology* phase. *Capturing traffic* can be done by the tool *dumpcap*[12], which is part of the *Wireshark*[13] 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

[8] http://www.aircrack-ng.org/
[9] https://github.com/sophron/wifiphisher
[10] https://www.archlinux.org/packages/core/any/netctl/
[11] https://github.com/byt3bl33d3r/MITMf
[12] https://www.wireshark.org/docs/man-pages/dumpcap.html
[13] https://www.wireshark.org/

of available tools based on performed manual experiments, further described in the thesis [20].

3 Attack Automation Using Developed wifimitm Package and wifimitmcli Tool

The implemented tool is currently intended to run on *Arch Linux*[14], 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 the package would work as a software library. Schema of the *wifimitm* package is in Fig. 2.

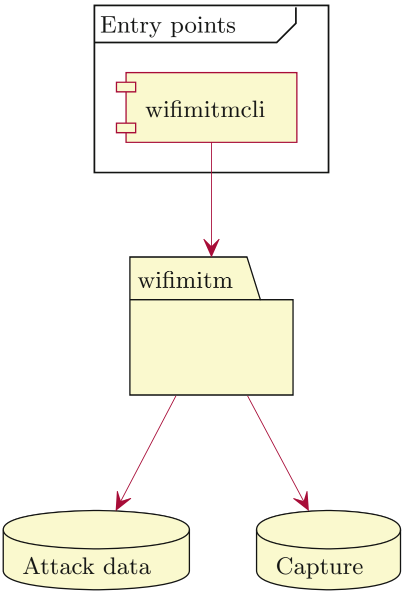

Fig. 2. This figure shows the basic structure of the developed application. The tool *wifimitmcli* uses a functionality offered by the package *wifimitm*. The package is also able to manipulate attack data useful for repeated attacks and capture files with intercepted traffic. Detailed structure of the package is described in Sect. 3.

The *wifimitm* package consists of following modules. The `access` module offers an automated process of cracking selected *WLAN*. It uses modules `wep`

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

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. 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`[15] 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. 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.

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*[16] 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.

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.

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

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

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 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

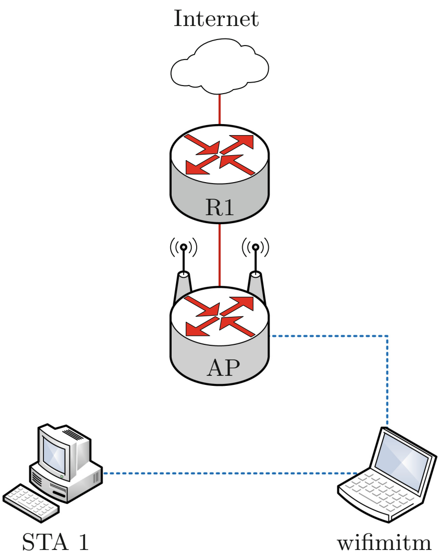

Fig. 3. This figure shows the network topology used for the first performance testing (Sect. 4) and success rate measurements (Sect. 5). Results of this performance testing are in Fig. 5.

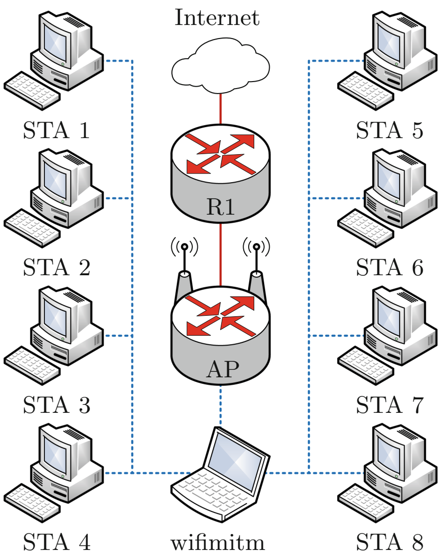

Fig. 4. This figure shows the network topology consisting of 8 *STA*s and 1 *AP* which was used for the second performance testing (Sect. 4). Results of this performance testing are in Fig. 6.

using the *aireplay-ng* tool. Managing monitor mode of interface is possible with the *airmon-ng* tool.

4 Attack's Performance Impact

A scheme of the networks used for the experiments is shown in Figs. 3 and 4. 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.

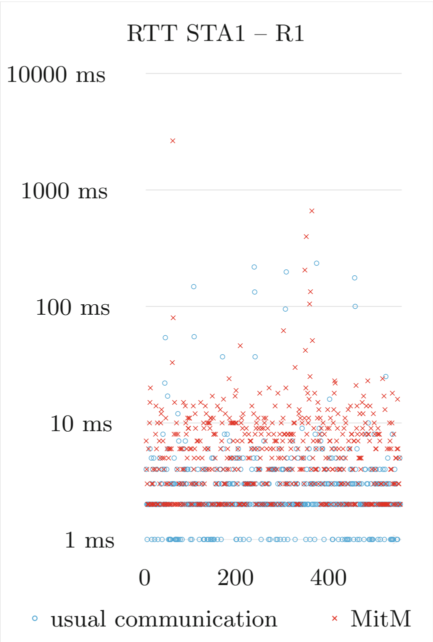

Fig. 5. The first *WLAN* for performance testing was the same as for the success rate measurements described in Sect. 5. Figure shows comparison of the measured *RTT* between *STA1* and *R1* during usual communication and during successful *MitM* attack. The results show the performance impact is not critical. Discussion with the users of the attacked network proved this attack unrecognizable.

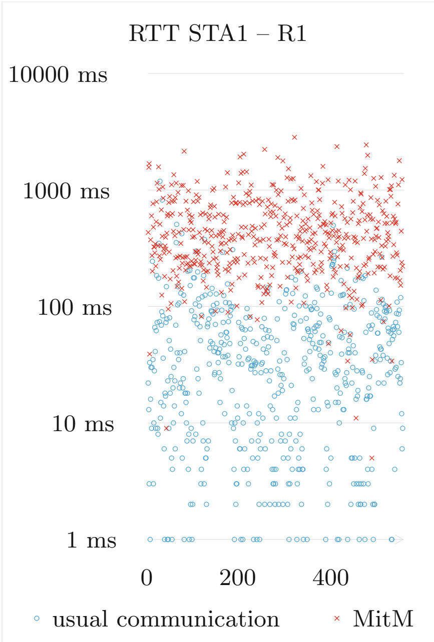

Fig. 6. The second performance testing consisted of 8 *STAs* and 1 *AP* connected to the Internet – streaming videos, downloading large files, etc. The figure compares the *RTT* between *STA1* and *R1* similarly. The performance impact is more severe than in Fig. 5. Despite the performance impact, the users had no suspicion that they were under *MitM* attack. Instead, they blamed the amount of devices for network congestion.

The performance impact of the *wifimitm* was compared using setups based on SOHO[17] environment. Both experiments were also evaluated based on the fact, whether the attack being performed was revealed or whether the users had any suspicion about the malicious transformation of their *WLAN*. Results of the testing are presented in Figs. 5 and 6.

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	✓	✓	✓	✓	✓

[17] Small office/home office.

5 Experiments Concerning Various Network Configurations and Devices

The test was considered successful if the *wifimitmcli* was able to capture network traffic according to the concept of *MitM*. For the test to be correct, no intervention (help) from the investigator was allowed during the attack performed by *wifimitmcli*. Results of the success rate measurements are shown in Tables 3 and 4.

Table 4. The following table shows the results of public experiments. Visitors of the Brno University of Technology, Faculty of Information Technology were invited to let their devices be attacked. Testing network utilized *Linksys WRP400* device as an AP. 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	✓

Results of experiments (Tables 3 and 4 and the thesis [20, pp. 42–43]) show, that 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). As Figs. 5, 6 and Tables 3, 4 show, *MitM* attack using the *wifimitm* is successfully feasible in the target environments.

6 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. 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 *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 Sect. 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 [15]. 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. Concerning the current state of the product, it does not focus on enterprise *WLAN*s, which also suffer from their own weaknesses.

The authors disclaim any use of this research for any unlawful activities.

References

1. Callegati, F., Cerroni, W., Ramilli, M.: Man-in-the-middle attack to the HTTPS protocol. IEEE Security Privacy **7**, 78–81 (2009)
2. Deal, R., Cisco Systems Inc.: The Complete Cisco VPN Configuration Guide. Cisco Press Networking Technology Series. Cisco Press, Indianapolis (2006)
3. Droms, R.: Dynamic host configuration protocol. RFC 2131, IETF, March 1997

4. Fluhrer, S., Mantin, I., Shamir, A.: Weaknesses in the key scheduling algorithm of RC4. In: Vaudenay, S., Youssef, A. (eds.) Selected Areas in Cryptography. LNCS, pp. 1–24. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45537-X_1
5. Godber, A., Dasgupta, P.: Countering rogues in wireless networks, vol. 2003-January, pp. 425–431. Institute of Electrical and Electronics Engineers Inc. (2003)
6. Halsall, F.: Computer Networking and the Internet. Addison-Wesley, Boston (2005)
7. Heffner, C.: Cracking WPA in 10 hours or less –/dev/ttys0 (2011). http://www.devttys0.com/2011/12/cracking-wpa-in-10-hours-or-less/
8. IEEE-SA. 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), pp. 1–2793, March 2012
9. Kent, S., Seo, K.: Security Architecture for the Internet Protocol. RFC 4301, IETF, December 2005
10. Klinec, D., Svítok, M.: UPC UBEE EVW3226 WPA2 password reverse engineering, rev 3. https://deadcode.me/blog/2016/07/01/UPC-UBEE-EVW3226-WPA2-Reversing.html. Accessed 5 Nov 2016
11. Klinec, D., Svítok, M.: Wardriving Bratislava 10/2016, 5 November 2016. https://deadcode.me/blog/2016/11/05/Wardriving-Bratislava-10-2016.html
12. Kumkar, V., Tiwari, A., Tiwari, P., Gupta, A., Shrawne, S.: Vulnerabilities of wireless security protocols (WEP and WPA2). Int. J. Adv. Res. Comput. Eng. Technol. (IJARCET) **1**(2), 34–38 (2012)
13. Liu, Y., Jin, Z., Wang, Y.: 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, September 2010
14. Plummer, D.: Ethernet address resolution protocol: or converting network protocol addresses to 48.bit ethernet address for transmission on ethernet hardware. RFC 826, IETF, November 1982
15. Pluskal, J., Matoušek, P., Ryšavý, O., Kmeť, M., Veselý, V., Karpíšek, F., Vymlátil, M.: 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 (2015)
16. Prowell, S., Kraus, R., Borkin, M.: Man-in-the-middle. In: Prowell, S., Kraus, R., Borkin, M. (eds.) Seven Deadliest Network Attacks, pp. 101–120. Syngress, Boston (2010)
17. Robyns, P.: Wireless network privacy. Master's thesis. Hasselt University, Hasselt (2014)
18. Tews, E., Weinmann, R.-P., Pyshkin, A.: Breaking 104 bit WEP in less than 60 seconds. In: Kim, S., Yung, M., Lee, H.-W. (eds.) Information Security Applications. LNCS, pp. 188–202. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-77535-5_14
19. Thomas, O.: Windows Server 2016 Inside Out. Inside Out. Pearson Education, London (2017)
20. Vondráček, M.: Automation of MitM attack on WiFi networks. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology (2016)