# Evil-AP - Mobile Man-in-the-Middle Threat

Kamil Breński, Maciej Choluj, Marcin Luckner

# Evil-AP - mobile man-in-the-middle threat

Kamil Breński, Maciej Chołuj, Marcin Luckner

Warsaw University of Technology
Faculty of Mathematics and Information Science
00–662 Warszawa, ul. Koszykowa 75, Poland
brenskik@student.mini.pw.edu.pl, cholujm@student.mini.pw.edu.pl,
mluckner@mini.pw.edu.pl

**Abstract.** Clients of public hotspots are exposed to various threats including a man–in–the–middle attacks. To stress existing threats we created the Evil-AP application for demonstrating a man–in–the–middle attack. The application, installed on an Android phone with root permissions, turns on hotspot services and performs network redirection. We tested as the proposed techniques can be used to eavesdrop, redirect, inject, and strip the Internet traffic. A mobility of the created solution together with the wide functionality creates an extremely dangerous tool. Therefore, we concluded our work with good practices that allow the users to avoid similar threats as described in our work.

## 1 Introduction

The users of mobile devices utilise various access points to connect with the Internet including public hotspots. They are exposed on various penetration tests [2] including man–in–the–middle (MITM) attack [12]. Such dangerous should be examined and a proper tools for that must be created.

In this work we presented the Evil-AP application developed for demonstration purposes. It sets up a MITM attack using an Android phone with root permissions, turns on hotpot services and preforms network redirection.

The performed tests showed that the proposed technique creates extremely dangerous mobile trap for careless public hotspots users. The Evil-AP can eavesdrop, redirect, inject, and strip the Internet traffic.

Similarly, as in works [13,3,11,10,9] the main aim of this document is to raise awareness of the dangers of MITM attacks. We demonstrated the methods that could be used by attackers, and provided appropriate methods and things to consider in order to stay safe, and keep our information private.

The rest of the paper is structured as follows. Section 2 contains information on a man-in-the middle attack and network protocols. Section 3 presents related works. Section 4 describes the Evil-AP application. Section 5 presents the tests of the application. Finally, Section 6 presents brief conclusions.

## 2 Preliminaries

A man-in-the middle attack intercepts communication between two systems. The focus of this work were Hypertext Transfer Protocol (HTTP) transactions. In this case the target is the Transmission Control Protocol (TCP) connection between a client and a web server. Using different techniques, the attacker splits the original TCP connection into 2 new connections, one between the client and the attacker and the other between the attacker and the server. Once the TCP connection is intercepted, the attacker acts as a proxy, being able to read, insert and modify the data in the intercepted communication. MITM attacks can also be done over Hypertext Transfer Protocol Secure (HTTPS) connections. The only difference consists in the establishment of two independent Secure Socket Layer (SSL) sessions, one over each TCP connection. The browser sets up a SSL connection with the attacker, and the attacker establishes another SSL connection with the web server. In general the browser warns the user that the digital certificate used is not valid, but the user may ignore the warning because he does not understand the threat. In some specific contexts it is possible that the warning does not appear, as for example, when the Server certificate is compromised by the attacker or when the attacker certificate is signed by a trusted Certificate Authority (CA) and the Common Name (CN) is the same of the original web site [8].

Since the developed Evil-AP proxy is an HTTP proxy, an overview of HTTP is presented. The HTTP protocol has always been a popular protocol [1]. It is the foundation of data communication in the world wide web. HTTP functions as a request-response protocol in the client-server computing model [7]. In the usual scenario a web browser like Firefox or Chrome is the client which makes the request to the web server for some particular resource like an HTML, CSS, or JavaScript file, along with any other resources needed to properly render a website. HTTP typically works over TCP which is a stateful protocol which provides a reliable connection. TCP is a transport layer protocol of the OSI model, while HTTP is an application layer protocol. For the purposes of this work we designed a HTTP proxy which will be between client and server communication. The HTTP protocol that is in common use today was developed in 1999 and is defined in RFC 2616 [4].

There are various requests that the client can make to the server, popular ones being the GET and POST requests. The GET request is used to retrieve remote data from a host server. Yet it is not uncommon to pass variables from one page to another by using the URL query string. On the other hand a POST request is used to insert or update remote data. Such data might include a comment or a block of data that is the result of submitting a web form. This data is sent in the body of the request and not clearly visible like variables in the URL of GET requests. There is a common misconception among new web developers that data sent in POST requests is safer than data sent in GET requests. That is not true as someone using a proxy can easily modify any part of the request, just like Evil-AP does. The data found in a POST request is quite commonly originating from a web form, this makes the contents of the body particularly interesting

to someone with malicious intent as it might contain personal information or login credentials. There are various other requests like the OPTIONS, HEAD, or TRACE requests but they are usually used less often. Evil-AP application supports the GET and POST requests.

## 3 Related work

Several works described MITM attacks. The high-level overview of what penetration tests are and what tools are used can be found in [2]. Some concrete examples of certain activities that might be performed during a penetration test are discussed, yet only a small section is devoted to MITM attacks. That section only describes MITM attack as Address Resolution Protocol (ARP) poisoning [5]. Work [12] identified the lack of a good taxonomy of MITM attacks against HTTPS. It established such a taxonomy in order to improve the SSL/TLS protocols by providing a better understanding of such attacks. It did not provide details on any concrete MITM attack implementation. Work [3] presented various security vulnerabilities that occur in WiFi networks. It focused on showing various ways that a malicious party can in fact monitor network traffic that various other legitimate users are generating. The work presented more attack types than MITM, yet it focused on sniffing private information like passwords of email and bank accounts, along with facebook activity, without editing any of the existing network traffic. The issue of encrypted traffic was not raised.

Several works discussed similar issues as presented in this work. We prepared a brief description of those works and stressed differences between our approach and other propositions. Work [13] presented a possible method of establishing a secure point-to-point connection using a shared secret derived from the fluctuations in a radio environment. Creating such a secure connection between two points would effectively render MITM attacks unlikely as someone who is not in close proximity to the two points that want to establish a connection will not have the same radio environment from which a token was derived. This work did not take into consideration the security and privacy of clients using a shared LAN network. Work [10] presented an epidemiological approach to simulate the spread of malware over poorly protected Wi–Fi Access Points. It focused on the aspect of compromising a LAN network, which gives the attacker the ability to perform MITM attacks. The work touched on the topic of compromising typical home routers and re-flashing them with new firmware which would give the attacker total control over the device. Its aim was to show the possible coverage and time needed in order to infect as many APs as possible. Work [6] presented a detection scheme for discovering Evil-Twin attacks on WiFi networks. The Evil-Twin attack is a certain type of MITM attack where the attacker poses as a legitimate Access Point and intercepts communication. Using appropriate hotspot configuration Evil-AP can also be used in an Evil Twin attack. Unfortunately the scheme discussed in this work only functioned in attacks that use a single ISP gateway, where Evil-AP uses a different ISP gateway, the one provided when using a mobile data plan. Extra steps are needed to ensure that the

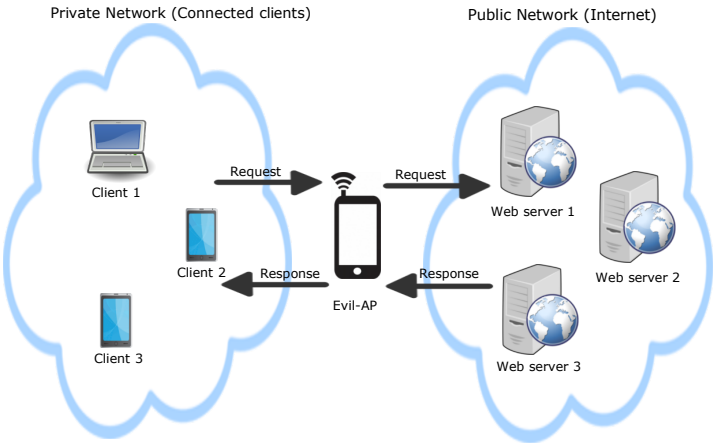



Fig. 1: Architecture overview.

detection scheme works for both scenarios. Work [9] demonstrated how Android applications can be exploited using MITM attacks with DNS spoofing. It also pointed out various security holes in the Android security model. Unlike Evil-AP this work used a laptop to set up a rouge access point, and focuses on editing the contents of a WebView that is displayed by an application. Finally, work [11] discussed a vulnerability found on Apple iOS devices, affected version being iOS 3 to iOS 5. The vulnerability occurs when an Apple device joins an open WiFi network, the device then automatically makes a request to one of the Apple's servers to test network connectivity. If the response is not what the device was expecting it makes the assumption that there is a captive portal and it automatically opens a UIWebView for the user to accept the terms of service. Problems begin when that response contains malicious content like a hook used to control browsers like the one that can be generated using the BEEF framework. This work focused on attacking client devices and not on security and privacy of data transferred on a LAN.

## 4 Evil-AP

At the core of the Evil-AP application there is a proxy server which listens for incoming HTTP and HTTPS connections. Furthermore this proxy can be seen as having two parts. The private network (LAN) facing part which will be a server that accepts connections from clients connected the hotspot and a public network (Internet) facing part which will send requests to web servers in our client's name. A request will have to go through our proxy server in order to be processed, logged, and perhaps modified by it. Similar rules apply to responses from web server. This scheme is represented in Figure 1. Internally the proxy

server is implemented as an android service which uses two thread pools, one for handling HTTP connections and the other one for handling HTTPS connections.

### 4.1 Applied Technology and Requirements

The Evil-AP application is implemented in the Java programming language using Android Studio IDE. Since the application also requires interaction with the underlying Linux kernel of the rooted phone, UNIX commands from within Java are used to accomplish certain tasks. These tasks include applying new firewall traffic redirection rules or reading certain system files in order to gather information about connected clients.

As the project requires at least a partial implementation of the HTTP protocol an external library is used to help with this. The library is called *okhttp* [14] and it is used to make requests to web servers.

The application requires the following permissions:

`ACCESS_CHECKIN_PROPERTIES` Allows read/write access to the "properties" table in the checkin database, to change values that get uploaded. Application requires these permissions in order to configure the hotspot properties.

`ACCESS_NETWORK_STATE` Allows application to access information about networks.

`CHANGE_NETWORK_STATE` Allows application to change network state.

`ACCESS_WIFI_STATE` Allows application to access information about Wi-Fi networks.

`CHANGE_WIFI_STATE` Allows application to change Wi-Fi state.

`READ_EXTERNAL_STORAGE` Allows application to read from external storage. Necessary for SQLite database to function properly.

`WRITE_EXTERNAL_STORAGE` Allows application to write to external storage. Necessary for SQLite database to function properly.

`INTERNET` Allows application to open network sockets.

The phone itself needs to give the application root permissions which allows it to redirect traffic using the Linux firewall called iptables. In order to do this the phone itself needs to be rooted. The specific rule used for HTTP redirection looks as such `iptables -t nat -I PREROUTING -i wlan0 -p tcp --dport 80 -j REDIRECT --to-port 1337` , a similar rule is used in the case of HTTPS. The application also uses the phones mobile data plan in order to connect to the Internet, this is because the wireless interface that could be used to connect to an existing WiFi is already being used by the hotspot. Another assumption that is made by the application is the fact that all requests made by the clients will have the host header present since it is needed in order to construct the okhttp request.

### 4.2 Evil-AP functionality

To describe the functionality of the Evil-AP application we defined two roles.

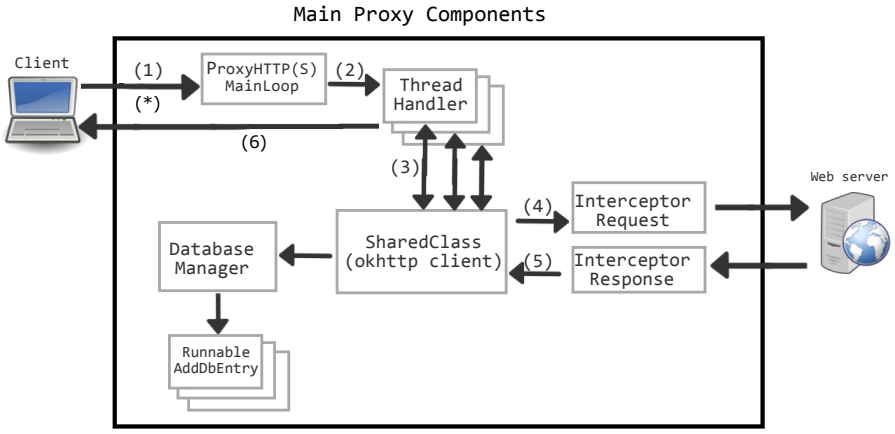



Fig. 2: Overview of main proxy components

The first role is the Evil-AP user. The user configures the hotspot. The configuration consists of a service set identifier (SSID), a password, and a security method.

Next, the user can define how the network traffic will be edited. There are three modes. In the first mode, the user selects the image to use as a replacement to all images in an HTTP response. In the second mode, the user writes the JavaScript code he wants to inject to an HTTP response. For security reasons, the JavaScript code is limited to an alert box and the user can only modify the presented text.

In the last mode, the user enables SSL Strip. In the last mode the application forces a client's browser into communicating in plain-text over HTTP. All https:// URLs are striped and turned into http:// URLs.

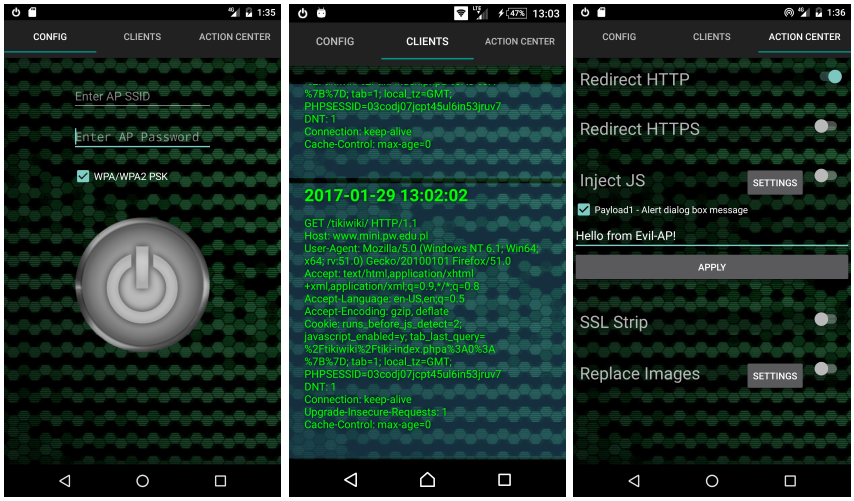
Additionally, the user can ban clients, view the client log and redirect HTTP(S).

The client, which is unaware, can only connect to the hotspot, browse the Internet, and disconnect.

### 4.3 Proxy Architecture

The core of the Evil-AP application is a set of proxy components. The components handle HTTP requests. The flow of a request through the proxy components is shown in Figure 2.

The client sends a request and it is rerouted to our application (1). Next,the `ProxyHTTP(S)MainLoop` accepts the connection and passes it to a thread from the thread pool (2). The `ThreadHandler` reads the client request and parses it into an okhttp request. While parsing the string request it uses the `SharedClass` object to insert new entries into the SQLite database. Internally `SharedClass` uses a thread pool to run SQL INSERT commands in order to avoid further delaying the handling of client connections. It then accesses the okhttp client found




(a) Configuration (b) Redirection (c) JavaScript injection

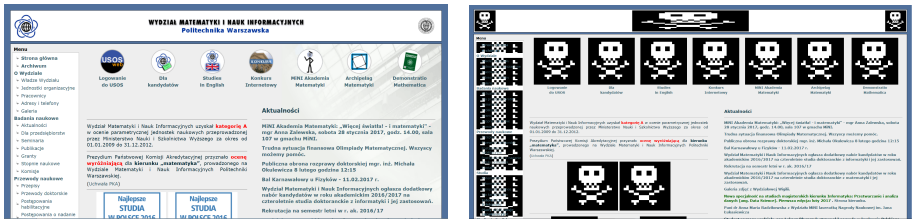
Fig. 3: Evil-AP configuration and tests

in `SharedClass` to make the request to the web server (3). Before forwarding the request to the web server `InterceptorRequest` removes any security related headers like "Upgrade-Insecure-Requests" or "Strict-Transport-Security" (4). The response from the web server is edited by `InterceptorResponse` . The status-line of the response has a hard-coded "HTTP/1.1" string in order to avoid using HTTP/2.0. This is also where various actions like injecting JavaScript or replacing the bytes of images take place (5). After receiving the response `ThreadHandler` forwards it to the client (6). When the client makes a regular HTTP request it will originally be destined for port 80. This scheme still applies if the client makes an HTTPS request, the only difference is that it will be destined for port 443 and before any data is exchanged the SSL/TLS handshake will take place. When that happens the proxy server will present the client with a self-signed certificate and the client should have an option to either add an exception for the untrusted certificate or be denied access to the web resource he is requesting (*).

## 5 Tests

In this section results of various Evil-AP configurations are shown. A single connected client was used to make a request for the faculty of Mathematics and Information Science website found at "http://www.mini.pw.edu.pl/tikiwiki". The Firefox browser was configured to not use its cache by changing entries.

The application was started as a hotspot and tested in several configurations as it is shown in Figure 3. The following configurations of Evil-AP were tested:




(a) rendered normally (b) with replaced images

Fig. 4: Faculty website

- Evil-AP not redirecting traffic (*no redirection*)
- Evil-AP only redirecting and logging traffic (*redirection*)
- Redirection with JavaScript injection (*js*)
- Redirection with image replacement and SSL Strip (*ir + ssl*)
- Redirection with image replacement, SSL Strip, and JavaScript injection (*ir + ssl + js*)

Evil-AP with *no redirection* did not establish a MITM attack between connected clients. The requests were not logged or edited. The hotspot worked normally, just as if Evil-AP would not have been installed. Figure 4a shows the faculty website rendered normally.

In the *redirection* configuration Evil-AP proxy established a MITM attack and logged client's requests. The requests were sent and the response was forwarded back without any editions. Figure 3b shows the logged initial request that was made by the client.

In the *js* configuration, the Evil-AP injected a small piece of JavaScript in the HTML response. The used script was `alert("Hello from Evil-AP!");` . As a result the alert box was displayed. Figure 3c shows configuration of the injection.

In the next *ir + ssl* configuration, the bytes of every image in the response were swapped, and all HTTPS links were changed to HTTP ones. All images were replaced by the image currently loaded in Evil-AP. The rendered website is shown in Figure 4b.

In the last configuration all features of Evil-AP were turned on. The user got a JavaScript alert and all images were swapped like in Figure 4b.

### 5.1 Performance Comparison

In order to test how the proxy impacts performance, 10 requests for the faculty website were made in each configuration. The amount of time for the initial request and the amount of time needed to render the entire website were recorded. For the final chart the average for those times was taken. To fully load the website 58 requests were made in total.

Figure 5 shows the performance comparison between various configurations of Evil-AP. It is visible that enabling various features causes some delay while

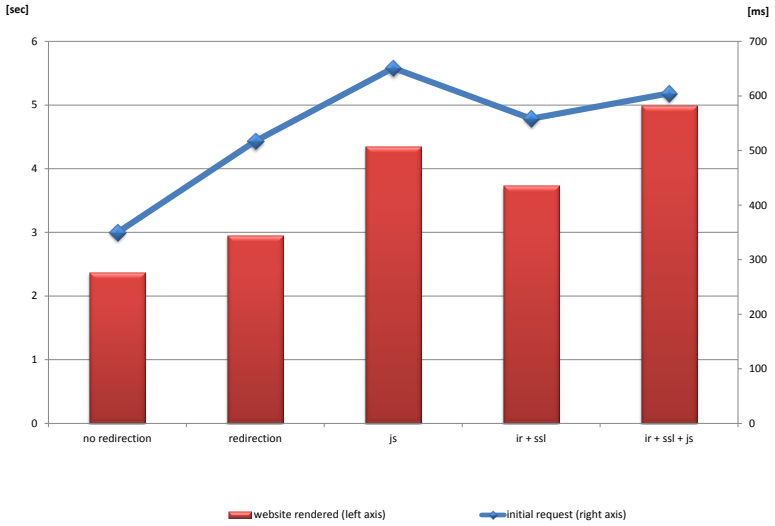



Fig. 5: Network performance for different configurations of Evil-AP

handling requests which results in more time needed in order to fully render a website. The delay is there yet it is not that significant mostly due to the fact that HTTP persistent connection is implemented inside the application. This allows the browser to use a single TCP connection to make multiple requests, without the overhead of setting up and tearing down each TCP connection for each individual request.

## 6 Conclusions

In our work, we have shown that the usage of untrusted access points can be extremely dangerous. Our mobile application Evil-AP can change, inject, or redirect HTTP and HTTPS traffic. The Avast free Wi-Fi experiment at Mobile World Congress[1] has shown that even experts are careless using unknown hotspots with a free Internet. And by its mobility, our solution is even more dangerous.

Maliciously performing MITM attacks like the ones discussed in this paper is not only wrong, it is illegal. The goal of this paper is not to enable attacks on hotspots clients, but rather to make people aware of the real dangers that exist to an individual's privacy so that better care can be taken in order to secure ourselves against these threats. We honestly hope that the information contained here will be used in good-faith by aspiring penetration-testers and security enthusiasts. According to our aim it is good to stress several good practises that

[1] https://blog.avast.com/2016/02/24/avast-free-wi-fi-experiment-fools-mobile-world-congress-attendees/

allow the users to avoid dangers connected with the attacks as one described in this work.

### 6.1 Good practises

Since a generated self-signed SSL certificate was used to set up a MITM attack on HTTPS connections for websites that do not use HSTS, a warning was presented to the client that there is something wrong with the connection. This attack relies strongly on the gullibility of the end-user. Therefore, the best thing to do is to never add any exceptions or add any untrusted certificates to the browser certificate store. If we do add an exception, then we should be perfectly aware of the real reason behind the message, or be prepared to deal with the very serious consequences that could result.

Using a VPN would circumvent traffic redirection entirely, in the case of Evil-AP. Since VPNs do not use standard ports, and no other rules exist other then the two that redirect traffic destined for port 80 and port 443, VPN traffic would never even be redirected to Evil-AP. Assuming that somehow it was. The authentication that needs to take place before any data is exchanged would fail. The worst case scenario being that the attacker could cut off connectivity with the VPN, which would be a clear indication that there is a proxy between us and the public Internet.

Modern browsers always give us an indication of what kind of connection is established with the host server. HTTPS connections are usually represented with a green lock somewhere around the address of the website we have connected to The lack of this green lock or a crossed out lock indicates a regular HTTP connection. This should raise our suspicion and we should be aware that anything we send or receive over this type of connection can be easily intercepted, monitored, or transformed.

In the case of the Evil-AP proxy there are a few indicators that could hint that network traffic is being intercepted. The biggest one being that the HTTP protocol is only partially implemented, meaning requests other than GET and POST will not work. This means that websites that rely on other methods will not render properly.

### 6.2 Possible improvements and future work

The Evil-AP application does not work in the case of servers configured with HSTS. This is because the browser has an entry saved that says it should only connect to this server using HTTPS and it stops the user from easily adding certificate exceptions for this host. While it is always nice for servers to be configured this way, it should not be treated as a final cure for MITM attacks. That is because a malicious actor could control the entire traffic flow, including DNS traffic. An upgraded version of SSL Strip can be therefore constructed. Instead of only changing all occurrences of https:// into http:// a small part of the hostname could also be changed, like adding an extra w in www, it would not really matter that such a host might not exists as the attacker is the one

that is doing the name resolution anyway. By doing DNS resolution on edited hostnames an attacker will be able to slip past already existing HSTS entries saved in the browser. Another possible improvement that would add potency to Evil-AP is the ability to inject forged 802.11 frames. This would allow for better execution of the Evil-Twin attack since the configuration of a legitimate AP can be copied, then the attacker would send forged 802.11 deauthentication frames with the spoofed address of the client that will be disconnected from the legitimate AP and perhaps re-connect to Evil-AP if the communication from the malicious AP is faster then the one from the legitimate AP.

## References


1. Dave Raggett, Arnaud Le Hors, I.J.: HTML 4.01 Specification. https://www.w3.org/TR/html4/ (1999), [Online; accessed 28-January-2017]
2. Denis, M., Zena, C., Hayajneh, T.: Penetration testing: Concepts, attack methods, and defense strategies. In: 2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT). pp. 1–6 (April 2016)
3. Fahmy, S., Nasir, A., Shamsuddin, N.: Wireless network attack: Raising the awareness of kampung wifi residents. In: 2012 International Conference on Computer Information Science (ICCIS). vol. 2, pp. 736–740 (June 2012)
4. Fielding, R.T., Gettys, J., Mogul, J.C., Nielsen, H.F., Masinter, L., Leach, P.J., Berners-Lee, T.: Hypertext transfer protocol – http/1.1. RFC 2616, RFC Editor (June 1999), http://www.rfc-editor.org/rfc/rfc2616.txt, http://www.rfc-editor.org/rfc/rfc2616.txt
5. King, J., Lauerman, K.: ARP Poisoning Attack and Mitigation Techniques. http://www.cisco.com/c/en/us/products/collateral/switches/catalyst-6500-series-switches/white_paper_c11_603839.html (2016), [Online; accessed 28-January-2017]
6. Nakhila, O., Zou, C.: User-side wi-fi evil twin attack detection using random wireless channel monitoring. In: MILCOM 2016 - 2016 IEEE Military Communications Conference. pp. 1243–1248 (Nov 2016)
7. ONE, I.: Hypertext Transfer Protocol. http://www.iotone.com/term/hypertext-transfer-protocol-http/t557, [Online; accessed December-2016]
8. OWASP: Man-in-the-middle attack. https://www.owasp.org/index.php/Man-in-the-middle_attack (2015), [Online; accessed 28-January-2017]
9. Park, M.W., Choi, Y.H., Eom, J.H., Chung, T.M.: Dangerous wi-fi access point: attacks to benign smartphone applications. Personal and Ubiquitous Computing 18(6), 1373–1386 (2014), http://dx.doi.org/10.1007/s00779-013-0739-y
10. Sanatinia, A., Narain, S., Noubir, G.: Wireless spreading of wifi aps infections using wps flaws: An epidemiological and experimental study. In: 2013 IEEE Conference on Communications and Network Security (CNS). pp. 430–437 (Oct 2013)
11. Spaulding, J., Krauss, A., Srinivasan, A.: Exploring an open wifi detection vulnerability as a malware attack vector on ios devices. In: 2012 7th International Conference on Malicious and Unwanted Software. pp. 87–93 (Oct 2012)
12. Stricot-Tarboton, S., Chaisiri, S., Ko, R.K.L.: Taxonomy of man-in-the-middle attacks on https. In: 2016 IEEE Trustcom/BigDataSE/ISPA. pp. 527–534 (Aug 2016)

13. Varshavsky, A., LaMarca, A., de Lara, E.: Enabling secure and spontaneous communication between mobile devices using common radio environment. In: Eighth IEEE Workshop on Mobile Computing Systems and Applications. pp. 9–13 (March 2007)
14. Wilson, J.: Okhttp wiki (May 2014), `https://github.com/square/okhttp/wiki`