

An Implementation of Client-Side Detection of Rogue Access Points

Filip Bellander

Oskar Zettervall

Department of Computer and Systems Sciences

Degree project 15 HE credits

Degree subject: Computer and Systems Sciences

Degree project at the Bachelor level

Spring 2014

Advisor: Irvin Homem

Reviewer: Henrik Boström

Swedish title: Att upptäcka vilda access punkter från klientensidan



Stockholm
University

An Implementation of Client-Side Detection of Rogue Access Points

Filip Bellander
Oskar Zettervall

Abstract

Wireless networks have become a popular convenience in restaurant, hotels, airports and many other places which brings the opportunity to perform work tasks, bank errands, and much more on the go. With this option also comes a danger. Anyone is able to set up their own network and let users connect to it. It is even possible to disguise it as an other network and trick users to connect to it and steal their information. This study implements and tests the algorithm proposed in Nikbakhsh et al. (2012). The algorithm is supposed to help the connecting client determine if the network comes from a legitimate access point or a rogue. It has however not been implemented and its validity has not been documented. This study aims to do this.

This study investigates how well (given as a percentage out of successful detections) the algorithm detects two types of rogue access points by first implementing the algorithm using Design Science and later test the algorithm using experiments. A version of the Man in the Middle attack, and a version of the Evil Twin attack was used to test the algorithm. The results were evaluated by taking all successful detections divided by the total number of performed attacks and a thorough analysis of the algorithm was conducted to explain the results.

The study found that the algorithm proposed in Nikbakhsh et al. (2012) was able to detect a very basic type of Man in the Middle attack and unable to detect an Evil Twin attack. Following the results, this study argues that the proposed algorithm should not be used and other alternatives should be investigated.

Keywords

Detecting Rogue Access Points, Client-side Detection, Wireless Security

Contents

Abstract	iii
List of Figures	vii
Glossary	viii
1 Introduction	1
1.1 Background	1
1.2 Access Points, Rogues and Legitimate	4
1.3 Man in the Middle attack	4
1.3.1 ARP spoofing	5
1.4 Evil Twin attack	5
1.5 Method Proposed in Nikbakhsh et al. (2012)	6
1.5.1 Textual description	6
1.5.2 Flowchart description	8
1.6 Problem	8
1.7 Research Question	9
2 Choice of Strategy, Method, and Analysis	11
2.1 Research Strategy	11
2.1.1 Mixed Methods	13
2.2 Research Method	14
2.2.1 Implementation Method	14
2.2.2 Test Method	15
2.3 Analysis Method	16
3 Implementing the Algorithm	19
3.1 Explicate Problem	19
3.2 Outline Artefact and Define Requirements	19
3.2.1 Small modification to the algorithm	19
3.3 Design and Develop Artefact	20
3.3.1 Infrastructure	21
3.3.2 Programming Language	21
3.3.3 Graphical User Interface	21
3.4 Demonstrate the Artefact	22
3.5 Evaluate Artefact	22
3.6 Development Conclusion	22
4 Execution	23
4.1 Tools Used	23
4.1.1 Access Points	23
4.1.2 Connecting Client	23
4.1.3 Attacker	23
4.2 Regarding chosen software	25

4.2.1	Wireshark	25
4.2.2	arpspoof	25
4.2.3	Aircrack-ng	25
4.3	Experimentation Set-up	25
4.3.1	Network Topology - Man in the Middle	26
4.3.2	Network Topology - Evil Twin	26
4.4	Executing the Experiments	26
4.4.1	Man in the Middle Attack	27
4.4.2	Evil Twin Attack	28
4.4.3	Ethical Considerations	29
5	Results & Analysis	31
5.1	Results	31
5.1.1	Man in the Middle	31
5.1.2	Evil Twin	31
5.2	Analysis	31
5.2.1	Man in the Middle	31
5.2.2	Evil Twin	31
6	Conclusion & Discussion	33
6.1	Conclusion	33
6.2	Discussion	33
6.2.1	Why it worked	33
6.2.2	Why it would not work in the real world	33
6.2.3	Flaws in the algorithm	34
6.2.4	The Algorithm, Simplified	37
6.2.5	Ethical Implications	37
6.2.6	Limitations of the Study	38
6.2.7	Future Research	39
Bibliography		i

List of Figures

1.1	Example of a Man in the Middle attack	4
1.2	Illustration of an Evil Twin attack	6
1.3	Rogue Access Point Detection algorithm as found in Nikbakhsh et al. (2012)	8
2.1	Overview of the Design Science Method as described in Johannesson and Perjons (2012;p. 36)	14
3.1	A modified version of the algorithm found in Nikbakhsh et al. (2012)	20
3.2	A screenshot of the CSWIDS application	22
4.1	Results from analysing the traffic from the client with wireshark, captured by the attacker.	24
4.2	Network topology used in the experiment	26
4.3	Network topology during the Evil Twin attack	27
6.1	Multiple levels of NAT	35
6.2	Simplified version of the algorithm proposed in Nikbakhsh et al. (2012) . .	38

Glossary

ACK	<i>Acknowledgement</i> A flag used in the TCP. If the ACK flag is set then the value of its field is the next sequence number that the receiver is expecting. This also acknowledges receipt of all prior bytes.
AP	<i>Access Point</i> A device that allows wireless devices to connect to a wired network using WiFi or related standards.
ARP	<i>Address Resolution Protocol</i> Used for resolution of network layer addresses into link layer addresses.
BSSID	<i>Basic Service Set Identifier</i> A unique identifier for a NIC that usually use the MAC-address as the identifier
CIDR	<i>Classless Inter-Domain Routing</i> A method for allocating IP addresses and routing Internet Protocol packets.
Client	<i>A client</i> is a physical object, a device such as a laptop, mobile phone, or any other kind of device that understands the 802.11 protocol.
DHCP	<i>Dynamic Host Configuration Protocol</i> A standardised networking protocol used on IP networks for dynamically distributing network configuration parameters, such as IP addresses for interfaces and services.
ETA	<i>Evil Twin Attack</i> See Section 1.4
ICMP	<i>Internet Control Message Protocol</i> Used by network devices, like routers, to send error messages indicating, for example, that a requested service is not available or that a host or router could not be reached.
IP	<i>Internet Protocol</i> An IP address is a numerical label assigned to each device participating in a computer network that uses the Internet Protocol for communication. The IP operates on the link layer.
IPS	<i>Internet Protocol Suite</i> The computer networking model and set of communications protocols used on the Internet and similar computer networks and is often referred to as the TCP/IP-stack due to its graphical representation if its abstraction layers looking like a stack with the TCP and IP being the most used protocols.
MAC	<i>Media Access Control</i> A MAC-address is a unique identifier assigned to network interfaces for communications on the network layer

MitM	<i>Man in the Middle</i> See Section 1.3
NAT	<i>Network Address Translation</i> Provides a method of modifying network address information in IP packet headers while they are in transit across a traffic routing device for the purpose of remapping one IP address space into another.
NetID	<i>Network Identifier</i> Part of an IP-address which identifies which network the address belongs to.
NIC	<i>Network Interface Controller</i> A computer hardware component that connects a computer to a computer network.
RSS	<i>Received Signal Strength</i> A measurement of the power present in a received radio signal.
SAS	<i>Short Authentication String</i> A method originally used in the ZRTP protocol where the SAS' are used to determine if the receiver is who she claims to be before divulging secret information.
Spoof	<i>-ing, -ed</i> A situation in which one person or program successfully masquerades as another by falsifying data and thereby gaining an illegitimate advantage.
SSID	<i>Service Set Identifier</i> Often used to refer to the name of a network.
TCP	<i>Transmission Control Protocol</i> One of the core protocols of the Internet protocol suit (IP). TCP provides reliable, ordered, and error-checked delivery of a stream of octets over a network.
TTL	<i>Time To Live</i> A numeric value that dictates how many devices an IP-packet may travel through. Decrementated at each device.
User	A person, a human being, utilising a client.

1. Introduction

1.1 Background

The Internet is arguably one of the greatest inventions of our time. It allows people to connect to each other and share ideas, expand their minds and learn new things. All the world's knowledge is within the reach of a few key presses. But to be able to take part in this information sharing we need to connect devices together. This is done with the help of networks. Networks consists of several devices (or smaller networks) which in turn get connected to other networks. The Internet is a network of networks (Kurose and Ross, 2012:p. 32).

In the beginning of networks, the most common way to connect devices together was by use of copper cables. This allowed users to send traffic to one another as long as the cable was connected to the device (given that the device was running and understood the needed protocols). This is still a common option, but the market has seen the introduction of newer types of cables (eg. fiber optics) and even the removal of the cable itself. When you remove the media through which you send your traffic (the cable), you need to replace it with something else. Before the idea of networks was born the use of radio waves was already in place since 1895 with Guglielmo Marconi's commercial wireless telegraphy system (Corazza, 1998).

In 1985, the beginnings of a new type of connection started to take form (Lemstra and Groenewegen., 2010). This connection was based on radio waves which would allow devices to communicate with one another without the need for cables, relying only on radio waves. The technology has developed into what today is often called WiFi (also spelt *Wi-Fi* or *Wifi*), which is the more common name for the 802.11 standard (IEEE, 2012). This standard has been implemented in a variety of devices such as laptops, tablets and phones.

The possibility to set up a network without the need for cables opens up the opportunity for both home and corporate users. By not needing cables, it has become easier to connect devices, in great or in small numbers, to the same network. This has also spawned a market for so called "hotspots" which are places where one can connect to a WiFi-network and connect to the Internet, sometimes paying a fee to get access. This set-up can be found in public locations like hotels, coffee shops, airports, etc. (ABI Research, 2013; Lemstra and Groenewegen., 2010:p. 154) Connecting to such a network, or any network using WiFi, comes with a risk(Godber and Dasgupta, 2003).

To tell clients that there is a network nearby a device is needed to broadcast this information. This is usually done with the help of an Access Point (AP)(IEEE, 2012:p. 5), a device which sends out a signal broadcasting the the presence of a network and information about how to connect to it. (Brain et al., 2001; IEEE, 2012:p. 44-91) Devices that receive this signal (called *clients*) can then follow the instructions and connect to the network. Sometimes there is a password that is needed to connect to the network and sometimes there is not, allowing anyone to connect.

Anyone is able to set up such a network, even with devices that are not dedicated to the task. This means that anyone could set up a network at a local coffee shop and let users connect to it. The owner of the network then has the power to do whatever she wants with the information being sent over the network. If the owner has malicious intent, she might eavesdrop on everything that is being sent, maybe even altering the

information. This is a typical case of an attacker setting up what is known as a “*rogue access point*”(Geier, 2003) that implements a “*Man in the Middle attack*”(MitM)(Kaufman et al., 2002:p. 167).

Another form of attack is known as the “*Evil Twin Attack*”(Bauer et al., 2008; Song et al., 2010). This is when the attacker broadcasts a signal advertising a network with the same name and authentication method as a preexisting one. A client will not be able to tell the networks apart and will think it is the same network. The attacker could then proceed to implement a MitM attack against any connected clients.

Network administrators have several ways to combat the presence of rogue APs. Shetty et al. (2007) describes a method in which a company may monitor all traffic going via its gateway router to try and detect if the origin of the traffic is from an authorised or unauthorised source. By analysing the traffic, the method can detect whether the traffic originated from a wired or a wireless source. It can then apply the suggested algorithms to detect whether the traffic originated from an authorised host or not and alert the network administrator if needed.

Ma et al. (2007) describes a system, which the authors call *RAP* (for **R**ogue **A**ccess **P**oint), that consists of three parts. The first part is a packet collector that listens to the traffic that is generated on the network. It also performs sniffing in a passive manner to gather even more information. The sniffing is done passively so as not to alert the attacker. The second part is a preemption engine. The researchers admit that network attacks cannot be avoided, but some can be prevented before they even happen. By analysing the data collected in the first part, the RAP system can take appropriate steps to find the attacker and prevent it from mounting the attack. The third part of the RAP system is a detection engine. This part detects rogue access points and any possible attackers in a reactive way. When an attack has occurred, the detection engine can help the network administrator deal with the situation. This third part is needed in extension to the preemptive engine as not all attacks may be recognised and a clever attacker could avoid the preemptive stage of the system.

Beyah et al. (2004) suggest using a temporal analysis of the shape of traffic sent in a network. By looking at the direction, time spacing, and amount of traffic in the network the researchers claim to be able to detect rogue access points at no more than one ‘hop’ away from the device doing the analysis. This method analyses the flow of information rather than the contents of that information, giving a different dimension to analyse for malicious content. With this analysis of the traffic the network admin should be able to detect a rogue access point should it connect to the network.

Yeo et al. (2004) provides research into pitfalls that need to be considered when using wireless LAN monitoring and suggest how these could be rectified. They define two important pitfalls that should be thoroughly considered by network administrators; (i) The sniffer used to monitor the traffic has fairly limited range with regards to both measurement loss and hearing (receiving). (ii) Carefully selecting the sniffers’ relative locations is important for acceptable capturing performance. The remedies suggestions are (i) Merging multiple sniffers data. This should give a better base for the analysis that will most likely follow the information gathering. (ii) Analyse where the best placements are for the sniffers. According to the authors of the papers, this is not close to the access point providing the network. By placing all sniffers at this position will cause the signal *from* the access point to overpower the sniffers ability to gather the signal going *to* the access point. The researchers suggest that only one sniffer should be placed near the access point and trying to place the rest near the wireless clients. By combining these two the network

administrator will have a good rate of information gathering for both information going *from* and *to* the access point (and hence, the network).

Wei et al. (2007) provides two different algorithms that uses TCP ACK-pairs to identify rogue access points. One algorithm requires training sets (data to determine what is normal behaviour) which has a detection rate of around 99%, and one algorithm without training sets with a detection rate of around 68%. The researchers also created an online testing system that implements both of these algorithms.

While all of these papers proposes ways that can detect rogue access points, they all have to be implemented by the network administrator. A user who intends to use the network does not know if any of these systems are in effect on a network. The user has no way to check the access point she is connecting to is safe or not using the above methods. These are all methods that the network administrator has to implement to ensure the users safety. There have however been some suggested way of bringing the possibility of identifying rogue APs to the client. Roth et al. (2008) describes a method where the client and the AP exchange *short authentication strings* (SAS) and displays the result of the exchange to the user with colours. The user presses a button and sees a colour. The user holds the button pressed for any length of time and if the colour changes at any time it might be the result of an Evil Twin Attack. There is however not possible to detect a Man in the Middle attack with this method. The MitM could just forward the traffic to the access point without altering it. The client would still get the correct information, but there is an eavesdropper in the middle that sees the whole process without the user being alerted.

Aziz and Hamilton (2009) describes a method to detect a Man in the Middle attack by timing. If Alice and Bob were to communicate with one another, message m_1 from Alice to Bob would take t_1 time to be received by Bob. If a MitM, Eve, were to intercept the message from Alice to Bob, the time to delivery would be greater than otherwise. Aziz and Hamilton (2009) describes a way to detect when this happens. A similar method to detect a rouge access point was described in Han et al. (2009). These methods are however limited to MitMs. They will, in their current implementation, not detect an Evil Twin. They measure the time it takes for the information sent to the recipient to be sent back. If the access point is an Evil Twin, there will be no difference than to a legitimate access point as the time to process the information will in all likelihood be the same. For example, if Alice sends a message to Bob, it will take the path Alice→Access Point→Bob. If Alice and Bob were connected to an Evil Twin, the message would take the path Alice→Evil Twin→Bob. The message would take a similar path as it would in a legitimate network and the timing-algorithm would not detect the attacker.

Nikbakhsh et al. (2012) describes a method which should be able to detect both an Evil Twin as well as a Man in the Middle. An algorithm was devised for this purpose that will, without the need for multiple authentication strings and message timing or otherwise needing to alter any existing protocols, detect these kinds of attacks. This means that unlike Roth et al. (2008) there is no need to have a predefined list of strings to send between the access point and the client. It also means that unlike Aziz and Hamilton (2009) and Han et al. (2009), there is no need to send several packets between the access point and the client and try to see a time difference, that might be due to other reasons than a Man in the Middle. Nikbakhsh et al. (2012) did however never publish any implementation of the algorithm proposed. If the algorithm is successful at detecting these attacks, it could be of great advantage to users who wish to be sure that an AP is a legitimate one and not a rogue.

1.2 Access Points, Rogues and Legitimate

An *Access Point* is a device that broadcasts a radio signal with information about a network (Gast, 2005:p. 14-16; IEEE, 2012:p. 5). A client that receives this signal can then authenticate against the AP and become part of the network. When connected, exchange of communication can be done as it would have in a wired network.

A legitimate AP is an AP that actually belongs to the network. It was the network owners intent for it to be there, and it will uphold the rules and polices set by the network owners. For example: A company may have a network to connect all their clients. All the APs that the company has set up to broadcast this network, and have not been tampered with otherwise, are legitimate ones. They do as the company's IT department have configured them to do.

A rogue AP is an AP configured with malicious intent. The owner of the AP aims to steal, alter, or otherwise compromise any information sent via the AP. The rogue AP may look like it belongs to a larger network, or it might stand by itself using a unique name and try to lure users in with promises of free Internet access (or such traits).

1.3 Man in the Middle attack

A Man in the Middle attack is where a malicious user positions herself between two parties. (Kaufman et al., 2002:p. 167) The two parties are unaware of the fact that a third party is reading and relaying their messages to one another. To demonstrate this, an example will be given.

Suppose Alice wants to send a message to Bob telling him to remember to pay her back for the lunch he owes her, giving him the bank account to transfer the money to. Then suppose that Trudy intercepts this message. Trudy now has the power to do what ever she wants with the message. She could forward it as is, or she might alter it and include her bank account instead of Alice's, or she might discard the message so Bob never receives it. Suppose that she sends the message on but has modified it before doing so. Bob receives the message and transfers the money he owes Alice to the bank account he just received. Bob makes sure to reply to Alice that he transferred the money to the bank account he was given and includes it in the message. Trudy receives the message, alters the message to include Alice's bank account instead of her own, and forwards it to Alice. Both Alice and Bob now thinks that Bob has transferred money to Alice's bank account. An illustration of this can be seen in figure 1.1

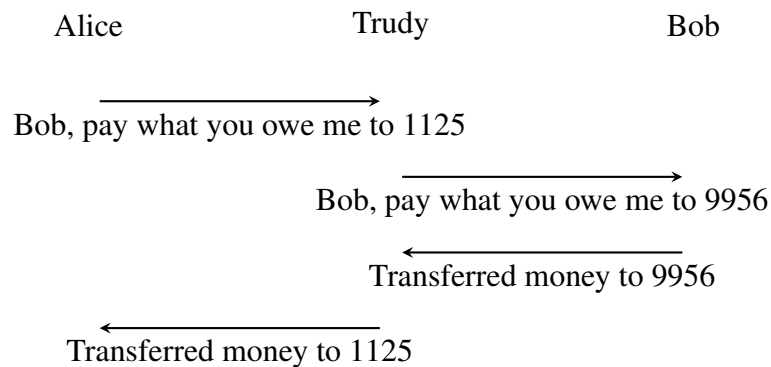


Figure 1.1: Example of a Man in the Middle attack

1.3.1 ARP spoofing

ARP spoofing is a technique that an attacker can use to send fake ('spoofed') Address Resolution Protocol messages into a network.(Lockhart, 2006:p. 184) By sending these counterfeit messages, the attacker can trick other clients to send IP-packets to the attacker instead of the intended recipient. When a host (client, router, switch, etc.) needs to send an IP packet to another host in the same network, the IP address of the recipient must be translated into the layer 2 MAC address. This is done with the help of the Address Resolution Protocol. The ARP translates layer 3 (IP) addresses into layer 2 (MAC) addresses, and vice versa. To find out what MAC address the owner of IP address 192.168.0.32 has, the requesting host sends out an ARP-request to all hosts in the network. The request, in a simplified form, looks like the following:

```
arp who-has 192.168.0.32 tell 01:d2:f3:cb:b5:cd
```

If a host is the owner of the IP address asked for, it replies with an ARP reply, which, simplified, looks like:

```
arp reply 192.168.0.32 is-at 11:22:4d:5c:ab
```

All other hosts should ignore the requests as they are not the owner of the IP address. Since the request contained the MAC address of the requester, the owner of the requested IP does not need to send out a request of its own, it already knows where to reply to. The requester receives the reply and is now able to send the IP-packet to the recipient.

What happens when a host that does not own the IP address requested but replies anyway? The ARP protocol has no built-in way to defend against this. The protocol is what is known as *stateless*, which means that it does not keep track of responses to the requests. This means that you can send out a reply even when there is no request! If a client were to start sending out a massive amount of ARP-replies claiming to be the owner of an IP which is not, the client would be able to intercept IP packets that was meant for another host. This is known as *ARP spoofing* and is a common technique used to intercept information destined for some other host, as is the goal of a Man in the Middle attack. ARP spoofing will be used in this study, see chapter 4.

1.4 Evil Twin attack

The Evil Twin attack is an attack where a wireless access point is configured to imitate a preexisting one in order to eavesdrop on network traffic. The attackers goal with setting up an access point that imitates another one is to steal information from anyone who connects to it. By looking like the legitimate access point, it can trick clients into connecting to the imitator, the Evil Twin, rather than the 'real' access point. Figure 1.2 shows a basic illustration of this sort of attack.

A wireless access point is identified by the "basic service set identifier" (BSSID) which is the MAC-address of the wireless interface in the access point.(Gast, 2005:p. 52) The BSSID is copied by the attacker and then given to the attackers wireless interface. The same is done with the wireless network name, service set identifier (SSID). (Song et al., 2010:p. 1)

By setting up an identical network (in the meaning of identification) the clients network manager will be lured and the two access points will appear as the same access point. The client will then use the one with the strongest signal strength. The attacker can move closer to the victim computer or transmit a stronger signal strength than the legitimate network.

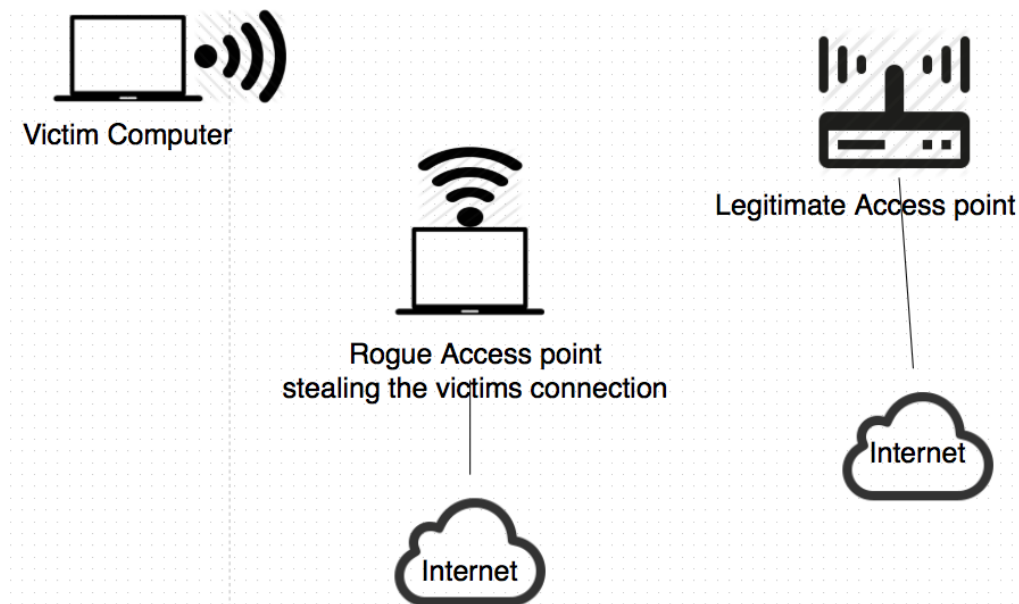


Figure 1.2: Illustration of an Evil Twin attack

The victim will not notice any difference in the connection (unless the legitimate network has a network storage, printer, etc., that the user uses) since the evil twin will route the traffic as usual to the Internet. The big difference between the legitimate and the rogue access point is that the rogue access point will be able to steal all the data packets going through the access point.(Song et al., 2010)

1.5 Method Proposed in Nikbakhsh et al. (2012)

Nikbakhsh et al. (2012) proposed a method that has the aim to help guide a user to a more secure network usage. When using a network it is very common that information is sent from the client to any possible destination. If this is not so, there is no reason to use the network at all as the purpose of a network is for clients to be able to communicate with one another.

The method that Nikbakhsh et al. (2012) describes in their paper implements an algorithm for detecting both Man in the Middle attacks as well as Evil Twin attacks. Nikbakhsh et al. (2012) suggested that the visual feedback to the user would be in the form of a traffic light, green for safe networks, yellow for untrusted, and red for unsafe. While this is a way to easily present the status of an access point to the user, it is not a necessity. This study will not use the traffic light feedback and only focus on the algorithm itself. Below follows a textual description of how it works with complementary reasoning, and later a graphical flowchart description of the algorithm.

1.5.1 Textual description

The client detects (at least) two APs that share the same SSID and BSSID (Basic Service Set Identifier(IEEE, 2012:p. 388)). It then proceeds to test these APs. In the first test, the IP addresses of the APs are tested. If they are equal, a trace route is performed on both APs. Nikbakhsh et al. (2012:p. 3) argues that the result of the trace routes could not be the same, as two APs could not share the same IP in the same network since it would cause an address resolution conflict. They have therefore chosen to mark this scenario as 'N/A'.

This means that the only possible result according to Nikbakhsh et al. (2012) is that the trace routes differ. Nikbakhsh et al. (2012) argues that this is the result of IP spoofing¹. Since the proposed algorithm cannot detect which AP has the spoofed IP, both are marked as 'Unsafe network'. Nikbakhsh et al. (2012) also notes that this should indicate that one of the APs is an Evil Twin.

The other possibility of the IP-test in the algorithm is that the IPs do not match one another. If the IPs do not match, the algorithm should calculate the Network Identifier (NetID) for the IPs. If the NetID is the same, it should indicate that the APs are part of the same network. The use of several APs is not uncommon in larger network as they are used for load balancing and to cover larger areas. Since this is a legitimate use of several APs, Nikbakhsh et al. (2012) argues that this means that both APs are safe to connect to and the user should be notified that they are such.

If the NetID is not the same however, the algorithm should execute a trace route on both access points and compare the results. The reason for this is to see if one of the APs has a longer trace route than the other. If this is the case Nikbakhsh et al. (2012) argues that this is a sign of an ongoing Man in the Middle attack with the following reasoning

"If there is any extra hop in the result, which is the proof of man in the middle, the red light will notify the user it is not safe to connect to this access point. In this state, the hacker had set up an access point to broadcast the same SSID as the public access point. The IP address of this network is different from the genuine one. The attacker lure users to connect to the rogue access point and after capturing packets they may pass them to the authorized access point. This will cause the extra hop in trace route result."

(Nikbakhsh et al., 2012:p. 4)

The last possibility in the algorithm is when both IP and NetID differs for the APs and the trace routes show different routes to the same destination. According to Nikbakhsh et al. (2012:p. 4) this happens when "*the attacker brings his own access point [...] and broadcast the same SSID.*" This of course means that the network is not one that the client should connect to. The algorithm can however not detect which is the AP that belongs to the network and which is the AP that the attacker has set up, both are therefore marked as 'Unsafe network' by the algorithm.

¹IP spoofing is when a device pretends to have a different IP than it actually has and is a common way for attackers to hide the real source of a IP packet. (Tanase, 2003)

and Aziz and Hamilton (2009); Han et al. (2009); Roth et al. (2008)’s suggestions for clients):

- Proposed method can detect both MitM and Evil Twin Attack.
- An end user can be warned of a rogue access point to prevent being exposed to the attacker.
- There is no need to modify the network architecture.
- End user mobile device can serve as detection mechanism.

Since the proposed method was never implemented and tested, it means that there are no results to judge the validity of the method by. The method might be a great resource for detecting rogue access points, or it might not work at all. If the algorithm works to some extent, testing the algorithm might provide insights as to how to improve it. It is therefore of great interest to investigate how well the algorithm actually performs so users will know if they should use it or not. As of the time of writing, no research has been done to investigate this.

1.7 Research Question

The research question is *“How well does the algorithm proposed in Nikbakhsh et al. (2012) perform in detecting rogue access points with respect to its described functionality?”*

2. Choice of Strategy, Method, and Analysis

The research question for this study is how well the algorithm proposed in Nikbakhsh et al. (2012) performs in detecting MitMs and ETAs. The algorithm has not been implemented and so conducting tests on it is not possible. There is neither any other research that evaluates, implements, or otherwise test the algorithm.

This chapter will discuss how this research proceeds in evaluating the algorithm.

2.1 Research Strategy

The authors of this research saw the following options of how to answer the research question:

1. Do a theoretical analysis of the algorithm and based on that evaluate how likely it is that it would succeed in detecting a rouge access point.
2. Implement the algorithm, conduct tests, and gather results regarding if the algorithm was able to detect a rouge access point.
3. Contact the authors of Nikbakhsh et al. (2012) and conduct interviews regarding its validity.

Each approach was then mapped to a corresponding research strategy:

1. Case Study
2. Mixed Methods
3. Surveys and Sampling

The approaches were then considered with the following reasoning:

Approach 1

The first approach would involve studying the algorithm and make assumptions about what would happen at different stages of the algorithm depending on theoretical constructs. This might lead to insights as to how well it works or how things should be made different. This approach would supply the researchers with the ability to come up with multiple different scenarios that could be studied that would could be hard to practically set up and test. This would also mean that, since there are more scenarios that could be tested, any possible flaw in the algorithm should be easier to see as more theoretical tests should go through it.

The downside of only conducting this theoretical approach is that it relies heavily on the researchers comprehension of the implemented network architecture in devices. While the IEEE (2012) specifies how devices should behave and what should happen when, it is not a guarantee that that is how it has been done in the real world. By not using a device the researchers will have to assume that the standard has been fully implemented, even if that is not the case. If there instead were practical tests, the researchers could be able to tell if the standard has actually been implemented as it should have been.

Not only must the researchers blindly trust in that the standard for WiFi has been fully implemented as suggested in IEEE (2012), the researchers must also have a complete and

correct understanding for it themselves. When using a practical test, the researchers can see the outcome of a test and don't have to pay much attention to the inner workings for a given component. When using theoretical tests however, the researchers must act as the components themselves, so to speak, and make sure the test behaves as such. This means that any test that is performed relies heavily on how well the researcher understands the given component. This results in a bigger risk taking as the researchers understanding might not, completely or in part, match the level of correctness as an implemented component might have.

The approach offers flexibility regarding the testing of the algorithm proposed in Nikbakhsh et al. (2012) as more test can be conducted without the need for any hardware requirements they might have. It does however require the researchers to fully understand the WiFi standard or the outcome will be incorrect. Another downside is that it would not solve the stated problem, that the algorithm was never practically tested. While the approach could give an indication of how well the algorithm would perform, it would still be theoretical, leaving the stated problem unsolved.

Approach 2

The second approach would involve first implementing the algorithm and after implementation conduct experiments to evaluate if the algorithm could detect a rouge access point.

The implementation would result in something that can be used and tested. It gives the researchers something to show to the world and, if the algorithm proves successful, recommend to users to make their network habits more secure.

The implementation phase also gives the researchers the process of fully immersing themselves in the algorithm to be implemented. As with any project, the developers (the researchers) need to fully understand what it is that should be developed for the project to be successful. This gives the researchers the opportunity to study the algorithm in depth and be able to pose questions about it that could be answered using tests in the second stage of this approach. It also borders on what could be considered a Case Study (Denscombe, 2010:p. 52). As the Case Study approach focuses on one or a few instances of a particular phenomenon, this accurately fit in with this approach. The authors will look at one specific algorithm, analyse it, and implement it. The authors will therefore get an in-depth understanding of the algorithm and be able to see possible faults in the proposed algorithm.

The second stage of this approach would be to conduct practical test on it. The practical tests would in this case mean using experiments. The researchers would propose two experiments that would test the algorithm for Man in the Middle attacks and Evil Twin Attacks. This will provide actual results about how the algorithm performs. It will however only provide results for those two experiments and the conditions they provide. Based on those results, it will however be possible to gather insights about how the algorithm performs as a whole and in other cases than those proposed for the experiments as long as the experiments are not too intricate and relies on specific settings and conditions.

This approach offers little flexibility. The researchers will have to implement the algorithm according to its description in Nikbakhsh et al. (2012), preferably using some form of Design Science. The researchers would then have to construct two experiments that will test the algorithms ability to detect a Man in the Middle and Evil Twin Attack. The experiments should preferably be as simple as possible as this should be a better indicator of how useful the algorithm is. By keeping it simple and not using advanced techniques

and the algorithm fails, one would have little hope that would do any better against more advanced attacks. It would therefore be wise to start with simple experiments and then move on to more advanced if the algorithm is successful in the more simple ones. It can then be shown that the algorithm could be used for simple attacks, but perhaps not so in advanced attacks. Using advanced attacks from the beginning would, according to the authors, probably result in simpler attacks not being tested.

The approach does have the advantage that it solves the stated research problem, by resulting in a testable artefact, as well as helping to solve the research question, by using experiments to test the algorithm.

Approach 3

The third approach would be to interview the authors of Nikbakhsh et al. (2012). The researchers would ask question regarding the choices the authors made when designing the algorithm. The researchers would then get the foundation on which the algorithm was designed on. This could give insights into when and how the authors envisioned the algorithm to be used.

While this method would give the researchers insights as to what the authors think about their algorithm, it would give little results and insights as to how well the algorithm fulfil its purpose. If the goal of the algorithm is to detect rogues, and it has not been proven that it can, why choose a research strategy that does not test this? Why choose a method that only investigates what the authors think about their own creation?

The interview approach has few, if any, merits that would make the researchers consider it a good candidate. While insights as to how the authors envisioned the algorithm to be used could be informative and a good basis for when recommending users to use it, it gives little of value in regards to its validity. This approach neither solves the stated research problem nor the stated research question. The approach is therefore not a valid alternative for this research.

Chosen approach

Out of the three approaches to answer the research question, approach number 2 is the only approach that solves the stated problem and answer the research question. With this reasoning, the used approach is to implement the algorithm and conduct experiments to evaluate how well it fulfil its stated goal.

2.1.1 Mixed Methods

As Mixed Methods was mapped to the selected approach, an explanation of how it will be used will be presented here.

Mixed Methods takes advantage of several different research methods to be able to answer the research question (Denscombe, 2010:p. 137). In this study it is necessary to both implement the algorithm proposed in Nikbakhsh et al. (2012) as well as conduct experiments to test it. The authors of this study therefore reason that one research method regarding the algorithms implementation and one research method regarding the testing of the algorithm should be used. This clearly separates the implementation-part from the testing-part, providing a more structured overview of the research.

The reason for separating the implementation and the testing, according to the authors of this study, will help possible future studies. It helps to show where mistakes could have been made and should make it easier to correct them. If the stages were to be bundled together, the authors argue that it would be more difficult to remedy the possible faults

as the process would have a more tangled structure. By separating the steps, the process becomes more linear and easier to pick apart should it be needed.

2.2 Research Method

As was reasoned in sections 2.1 and 2.1.1, two different methods should be used to structure the research and separate the implementation from the testing. This section will present the methods used in the study and the reasoning behind the choices.

2.2.1 Implementation Method

To implement the algorithm proposed in Nikbakhsh et al. (2012) a scientific approach is preferred over a non-scientific one, such as a ‘just-start-coding-it’-way. By using a scientific approach, each step of the development should contain a structured way for what should be done and what is needed. One such approach is the Design Science Method as described by Johannesson and Perjons (2012). Design Science offers a framework with a structure of activities to help develop an artefact meant to solve a problem. This study aims to implement an algorithm, which is to develop an artefact, to be able to conduct tests on it. The problem this study stated was that there is no such implementation to conduct tests on. By using Design Science to develop an artefact that implements the algorithm, the research problem is solved. The research question will also have an artefact to conduct its test against and it will be possible to evaluate if the algorithm can detect rogue access points.

Design Science provides several steps to be performed in a given order, as can be seen in figure 2.1. By following the steps outlined the method gives a structured and planned overview of what should be done when. It was therefore reasoned that by utilising Design Science, the development phase of the study would result in a more accurate implementation and a clearer overview of what was done during this stage.

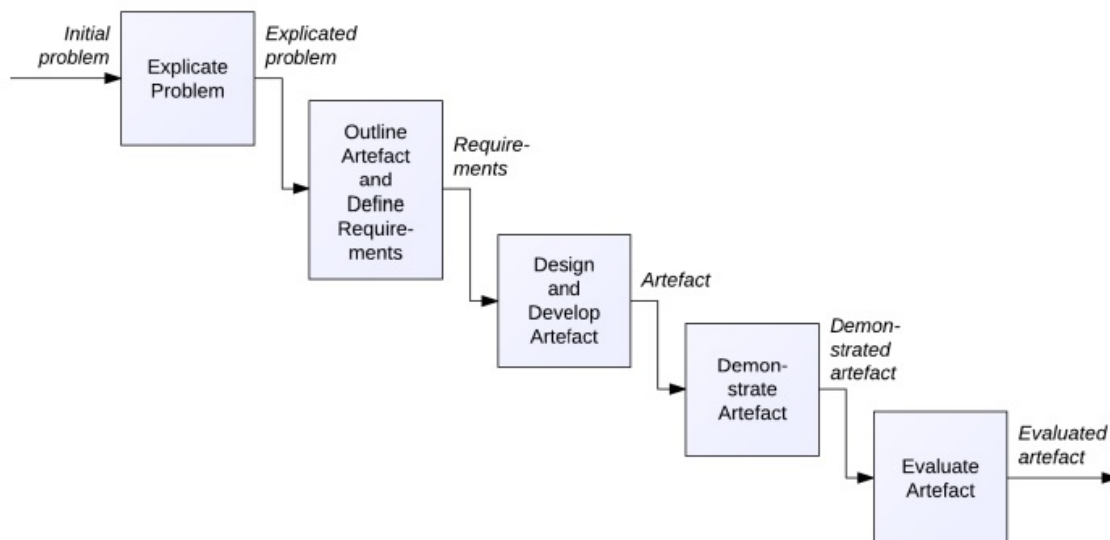


Figure 2.1: Overview of the Design Science Method as described in Johannesson and Perjons (2012:p. 36)

As the authors will need to fully immerse themselves in the algorithm and analyse each step of the algorithm to be able to implement it, it could be argued that the authors will then perform a form of a Case Study(Denscombe, 2010:p. 52). As the Case Study approach focuses on one or a few instances of a particular phenomenon, this accurately fit in with this study. The authors will look at one specific algorithm, analyse it, and implement it. The authors will therefore get an in-depth understanding of the algorithm and be able to see possible faults in the proposed algorithm. These possible faults should not go undocumented as they are a vital part of understanding how to improve the algorithm. The authors have therefore chosen to include these findings in chapter 6. They will also help understand why the results gathered from this study are as they are.

2.2.2 Test Method

To be able to test the implemented algorithm, there has to be a defined way to perform the tests. The authors of this study have therefore chosen to set up two experiments, found in chapter 4. These experiments are needed to be able to test the algorithm a theoretical infinite number of times in the same way. By doing the same thing over and over, it can be proven with a higher degree of certainty that the results are correct.

Each conducted experiment gives a result. To gather the results the authors of this study have chosen to use Observations(Denscombe, 2010:p. 196). By using systematic observations it will be possible to gather the results from the experiments in a simple manner. Observations also give results which are ‘pre-coded and ready for analysis’(Denscombe, 2010:p. 204) which is another reason for why it is suitable as it will eliminate a possibility of misreading the results.

Systematic observations lend themselves well to this research experiments. The research uses experiments in a quantitative manner. One experiment should be conducted several times. By using systematic observation the results will be gathered in an easy manner as the results from the experiments can be easily categorised. This also helps during the analysis stage as it is often easier to analyse predefined categories than analysing data that are in free-form.

2.2.2.1 Test cases

Section 2.2.2 mentions that two different experiments will be used to be able to test the implemented algorithm. This section will describe how these experiments were thought up.

Nikbakhsh et al. (2012) claims that the algorithm will help detect two different types of attacks, Man in the Middle and Evil Twin. To test this, there is a need for two different experiments as these are two different forms of attacks.

Experiment 1: Man in the Middle

The Man in the Middle(see section 1.3) experiment is aimed to be as simple as possible. The most basic form of a Man in the Middle attack that the authors can think of is with the use of arpspoofing. By simple telling the client to send its traffic through the attacker, the attacker has full access to everything that is sent from the client. The reason for performing this type of attack is because the low level of sophistication needed for it. By repeatedly sending ARP-packets to the client telling it we are the gateway, we can see all the traffic that it is sending. By also performing the same attack against the gateway the attacker

also sees all traffic going to the client. In short the attacker sees everything the client does without the need of advanced techniques.

Alternatives to this, that does not involve making the experiment more complicated, would be other types of attack that would lure the client to send its traffic via the attacker in different ways. As this approach uses the ARP protocol, an alternative method would have to use a different protocol or operate on a different layer of the IPS model to be of real difference. The authors of this study can think of no common or easy alternative to this. The authors can come up with an attack that would trick users into connecting to a fake webportal, as can often be found on ‘hotspots’, where you have to sign in to be able to use the network and then route any traffic via that webportal. This would however be much harder to set up and is more of an phishing(Dhamija et al., 2006) attack or an Evil Twin attack than a Man in the Middle attack, which is what the algorithm should test for.

Experiment 2: Evil Twin Attack

The Evil Twin Attack is used to trick clients to connect to a malicious access point instead of a legitimate one. This type of attack is often performed where there is a network already present that the attacker can imitate, luring users to use the malicious twin instead of the legitimate network.

The authors see two possible scenarios for how an attacker would act during an ETA. The attacker either sets up her twin and lets users connect to it in their own time, or she makes sure to also disconnect them from the legitimate network. By disconnecting already connected clients the attacker not only get more targets, she could also have the advantage that users will not bother to check if the new connection is safe or not. The users could just continue their network usage as before they were disconnected. The authors argue that it is more likely that an attacker would opt for disconnecting clients and force them to use her network. This also reduces the time the attacker needs to gain information, and time could be an important factor. As the attacker in all likelihood would want to limit her footprint, giving administrators and law enforcement less time to act and detect the attacker, she would probably try to have her twin active for as short a time as possible. This, according to the authors, results in that the attacker would disconnect clients from the legitimate network.

An Evil Twin Attack can be done in many ways. The main form is that an attacker sets up a ‘hotspot’ that appears to be genuine. Users then connects to the network and unwittingly gives away all their information. The Evil Twin does not strictly have to be a Twin, the attacker could set up the network where there are no other networks . The ETA then becomes a ‘rogue network’. It is however common to set them up where there are networks already present, hence the name.

As for how an ETA is actually implemented, there are several ways. The most common way involves copying information from a legitimate network and sending out this information from the Twin, masquerading as the legitimate network. The attacker then disconnects clients from the legitimate network, tricking them into connecting to the Twin. The attacker now has access to any information sent to and from the client.

2.3 Analysis Method

As this study tries to answer the question of how well the algorithm proposed in Nikbakhsh et al. (2012) perform in detecting rogue access points, it is necessary to specify how this result will be given. The result is not meant as a subjective one with the authors stating that X is better than Y, rather it is meant as an objective one. The analysis

used will therefore use the simple mathematical formula of

$$\% = \frac{SuccessfullyDetectedAttacks}{NumberOfAttacks}$$

This will give a percentile success rate of the algorithm in detecting rogue access points. By taking the number of successfully detected attacks divided by the total number of performed attacks, a simple answer to how well the algorithm performs will be given. The same formula could be used for any alternative method, giving a generic method for method-evaluation.

Based on the results from the experiments it will be possible to determine if the algorithm has any merit in detecting rogue access points. If the algorithm fails to detect the rogue for the majority of the time (ie. less than 50% success rate), the authors of this study would deem the algorithm unfit for general use. If the reverse is found (greater than 50% success rate), the authors would recommend the general public to use the algorithm as it would be a useful tool to help combat rogue access points.

Saying that a method is ‘good’ when it works only 51% of the time could be dangerous. It could mean that users use the method as a false security-blanket. If users come to trust the method, it can lead to users thinking that they are safe from rogue access points when in reality, they are not. The method only works *most* of the time, not always. Therefore, saying that a users should use it if it works 51% of the time comes with a danger. The authors do however argue that 51% of the time gives better security than no security at all. The same could be said for 10% of the time, but with this low success rate the users are more likely to get false positives (as in, the algorithm says it is safe when it is not) than true positives.

As for the analysis itself, the authors could not think of a method that would do the algorithm more justice or convey the true results of the experiments any better. The experiments provide results regarding how well the algorithm performed. When dealing with a question of the type “Did X do Y?” it often becomes a boolean question. Either X did do Y, or X did not. The answer can be elaborated as to *why* X did or did not do Y. This analysis method is however meant to investigate the boolean value of the question. It is therefore difficult, according to the authors, to come up with alternative analysis methods that would be as accurate as the one provided.

The reason for the results will however be of interest as this could help improve the algorithm if needed. Why X did or did not do Y will therefore be discussed in chapter 6.

3. Implementing the Algorithm

This chapter discusses the design and development of the application that will implement the algorithm proposed in Nikbakhsh et al. (2012). As was said in section 2.2.1, Design Science will be used to develop the artefact that implements the algorithm. Following will be a description of what was done at each step of the framework provided by the Design Science Method. The steps can be seen in figure 2.1.

3.1 Explicate Problem

An algorithm has been proposed in Nikbakhsh et al. (2012) that, according to its authors, can detect rogue access points from the connecting client and also be able to detect if there is a Man in the Middle or an Evil Twin attack in progress. The algorithm was however never implemented. Any practical tests on using the algorithm have not been documented. Only theoretical analysis have been proposed by the authors themselves.

3.2 Outline Artefact and Define Requirements

An artefact is needed to be able to test the algorithm proposed in Nikbakhsh et al. (2012). The artefact should implement the algorithm as is described in Nikbakhsh et al. (2012) unless deemed not possible or inefficient. For example, conducting the trace route at the same time as looking up the IP of the access point might be more efficient than first connecting to both access points and compare the IPs only to find that a trace route is actually needed and thereby having to reconnect to the access points one more time. The tests posed in the algorithm must however be done in the proposed order, even though the results used by the algorithm may have been gathered ahead of time.

3.2.1 Small modification to the algorithm

A problem was discovered in the algorithm during implementation. While it is true that it could be a sign of an Evil Twin attack when two access points share the same BSSID, a client would never see the access points that share the same BSSID. If the network name, the BSSID, and the broadcasting channel is the same, then the the client will see them as one and the same access point. Due to this, the algorithm would never be triggered. The algorithm requires two access points to be tested. It is indeed possible to set up two access points and make them send out the exact same information. The client, who is not able to detect anything but a radio signal flying in the air, would however only see one radio signal that carries that information, even if it is coming from two sources. The client is therefore unable to use the two sources for testing.

To be able to use the algorithm at all, it would require that the BSSID must be allowed to be different. This would allow two access points that share network name and broadcasting channel to be tested against one another. The result of the modification of the algorithm can be seen in figure 3.1. Since it is possible that two access points have the same IP and the trace route could be the same, the outcome that was previously described as N/A is now a possible outcome. It has been marked as 'Possibly Unsafe' since the access points could be used in a legitimate way (ie. load balancing) or it could be an attacker.

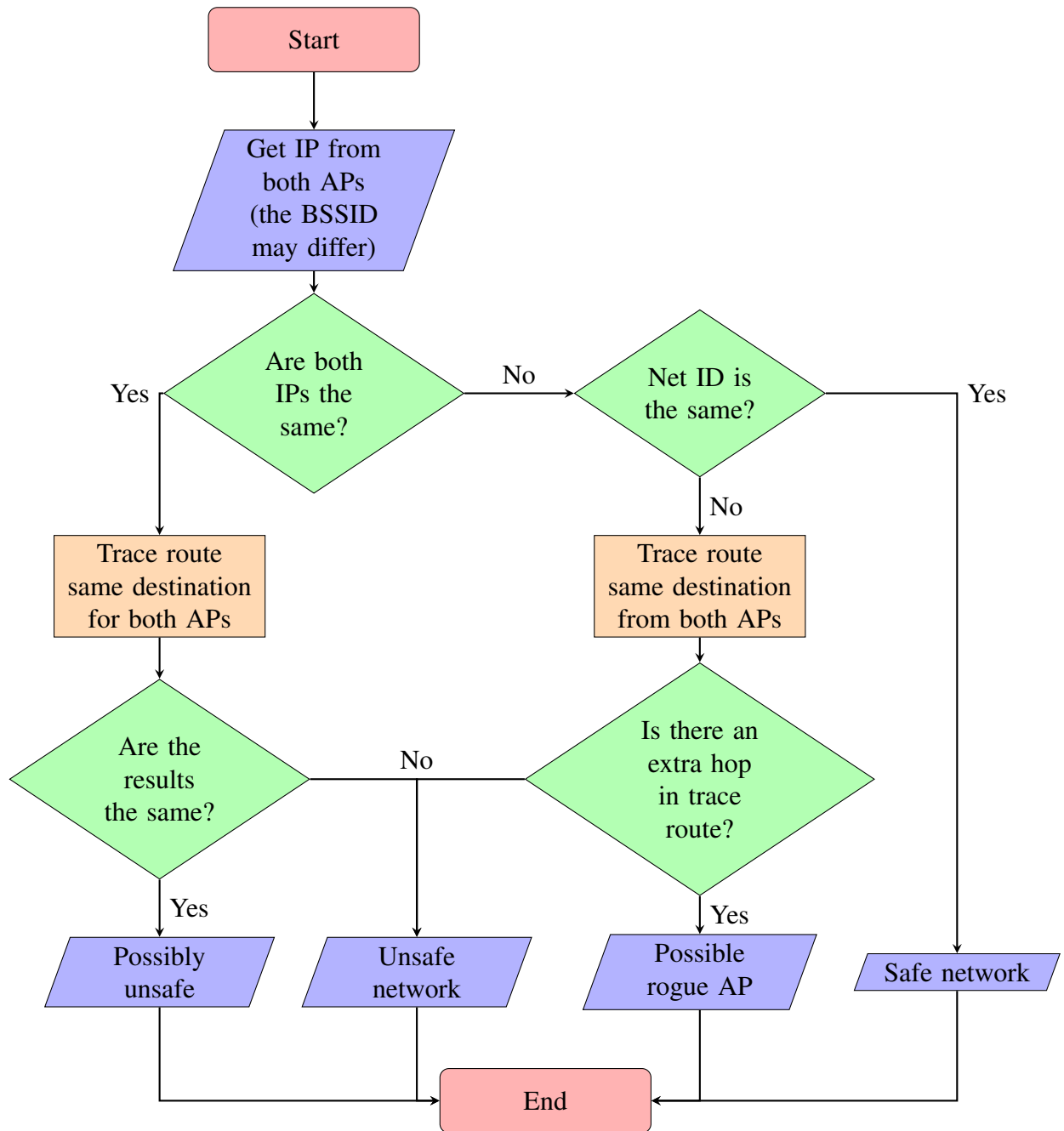


Figure 3.1: A modified version of the algorithm found in Nikbakhsh et al. (2012)

3.3 Design and Develop Artefact

In this section, it is explained what infrastructure was used to develop the artefact, what language and libraries were used, and how the algorithm was implemented.

The artefact that was developed was a GTK+¹ desktop application written in Python² 3. The name of the application was CSWIDS, which stands for **C**lient-**S**ide **W**ireless **I**ntrusion **D**etection **S**ystem.

¹<http://www.gtk.org/>

²<https://www.python.org/>

3.3.1 Infrastructure

To keep the development as open as possible, a github³ repository was created to be able to share the resulting artefact as easy as possible. It also made collaboration of the development easy as the source files were kept in a repository instead of at one local computer.

The repository used can be found at <https://github.com/qwelyt/CSWIDS> and the source code is published under a GNU GLPv2 licence so it is open for modification by anyone.

3.3.2 Programming Language

The programming language used to create the application was Python 3. The language was chosen for its ability to use native programs on the host system. For example, to use the GNU/Linux native program `iwlist` to scan for available wifi-networks with python, a single line is needed. In the terminal, the command would look like `iwlist wlan0 scan`. The result would be all networks that the network card associated with the `wlan0` interface could find. In python, this can be done using the `os.popen()` function. The line `networks = os.popen("iwlist wlan0 scan")` would execute the command and put the result in the variable `networks`.

It was also chosen for its ability to use the GTK+ library which allows for ease of porting across different platforms. The GTK+ library provides a graphical interface for the user, making it easier to select which network card to use, what access points to test, and to what destination the trace routes should be directed.

Other languages that were considered were Java, C, and C++. These languages were either not chosen because of restrictions in the language, as seen by the researchers, or the researchers own limited knowledge of the language.

3.3.3 Graphical User Interface

As the algorithm could prove to be a success at detecting rogue access points, it would be good if users took the habit of using it. To implement the algorithm in an easy-to-use application right from the start would be beneficial for this purpose. If all a user has to do is click on a few buttons, as opposed to writing a command line, the use of it will probably be greater than otherwise. With this reasoning in mind, a graphical interface was developed to help the user use the application. While a strict command line interface could have been used by the researchers, a non-technical user would probably not use such an application.

How the application looked can be seen in figure 3.2. The user first selects the network card to use in a drop-down list. The user then presses 'Scan' to scan for networks. It is now possible to select two networks from the list that has appeared. The address to test against can be changed if the users wishes, but at this point it is now possible to press 'Test' and the selected networks will be tested. The users will see the results in the lower part of the window. Figure 3.2 displays the window just after a scan has been performed.

³<https://github.com/>

The screenshot shows the CSWIDS (Client-Side Wireless Intrusion Detection System) application. At the top, there's a title bar with the application name and a close button. Below the title bar, there's a header area with a dropdown menu set to 'wlp1s0', a 'Scan' button, a 'Test site:' field with 'dsv.su.se', and a 'Test' button. The main area contains a table with the following columns: ESSID, Strength, Encryption, BSSID, Channel, Frequency, Bitrates, and Mode. The table lists six detected networks. At the bottom, there's a status bar that says 'Scan initiated... Done!'.

ESSID	Strength	Encryption	BSSID	Channel	Frequency	Bitrates	Mode
Cavejohnsson	91	WPA	02:25:9C:C0:83:C1	6	2.437 GHz	1, 2, 5.5, 6, 9, 11, 12, 18, 24, 36, 48, 54,	Master
GlittrigaRosaMoln	90	WPA2	D8:50:E6:C6:E1:24	13	2.472 GHz	1, 2, 5.5, 6, 9, 11, 12, 18, 24, 36, 48, 54,	Master
Aperture	85	WPA2	00:25:9C:C0:83:C0	6	2.437 GHz	1, 2, 5.5, 6, 9, 11, 12, 18, 24, 36, 48, 54,	Master
good	85	None	02:25:9C:C0:83:C2	6	2.437 GHz	1, 2, 5.5, 6, 9, 11, 12, 18, 24, 36, 48, 54,	Master
bad	84	None	02:25:9C:C0:83:C3	6	2.437 GHz	1, 2, 5.5, 6, 9, 11, 12, 18, 24, 36, 48, 54,	Master
Vale	78	WPA2	2C:B0:5D:AA:30:5F	1	2.412 GHz	1, 2, 5.5, 6, 9, 11, 12, 18, 24, 36, 48, 54,	Master

Scan initiated... Done!

Figure 3.2: A screenshot of the CSWIDS application

3.4 Demonstrate the Artefact

The artefact was demonstrated by using it in the experiments used in this study, see chapter 4. This solves the explicated problem of not having an implementation of the algorithm proposed in Nikbakhsh et al. (2012).

3.5 Evaluate Artefact

How the developed artefact actually performs can be found in chapter 4 and 5. This section will evaluate how well the artefact was developed in regards to the original algorithm.

The algorithm was implemented as described in Nikbakhsh et al. (2012) for the exception of the requirement that the BSSID must match, as discussed in section 3.2.1, and when the trace routes were made. Instead of first connecting to both access points, retrieve the IP of the access point, and then compare the IP only to risk face the need to connect to the access points again to perform a trace route, the trace routes were performed directly upon connecting. The results were then stored and the algorithms steps were followed dutifully.

3.6 Development Conclusion

In the above described way, an application was created that implemented a slightly modified version of the algorithm proposed in Nikbakhsh et al. (2012). Throughout the development the application was tested to make sure that it could perform the task that was needed from it. After the completion of the application, it was used in the experiments described in chapter 4, the results are presented in chapter 5.

4. Execution

In this chapter the experiments will be explained. There will also be a section regarding the ethical aspects surrounding the experiments after the the description of the experiments. How they were set up and performed is explained below.

4.1 Tools Used

Following is a description of the tools used to conduct the experiments.

The chosen hardware was not selected due to any specific reason. They were simple the hardware the researchers had available. Any other similar hardware should suffice to recreate the experiments.

4.1.1 Access Points

Two access points were needed for the experiments to be able to test the validity of the algorithm. If two access points share the same SSID but the IP differ, the algorithm must investigate if they share the same NetID, as can be seen in figure 3.1. If they do, they belong to the same network and are used for load balancing. For this purpose, two access points were needed. The hardware used were

- Linksys WRT54GL
- Linksys WRT320N

These access points were set up to broadcast the network “*RogueAPTtesting*” (see section 4.4.3 *Ethical Considerations* for the choice of name). They were configured to be open, meaning that no password was needed to connect, and were connected to the same switch to make sure they were connected an equal number of hops away from the Internet connection. No other configuration was done to the access points.

4.1.2 Connecting Client

The connecting client was an Asus eeePC 901 running ArchLinux¹ 32bit which was last updated at 20-04-2014 with the latest drivers and systemfiles.

The client was running the CSWIDS program (see chapter 3) to connect and test the algorithm.

4.1.3 Attacker

The attacker was a Dell Latitude E4200 running a live distribution of Kali Linux² 1.0.6 32bit.

The algorithm proposed in Nikbakhsh et al. (2012) should be able to detect two types of rogue access points, those who are Evil Twins and those that implement a Man in the Middle attack. What was used in each of the attack types is described below.

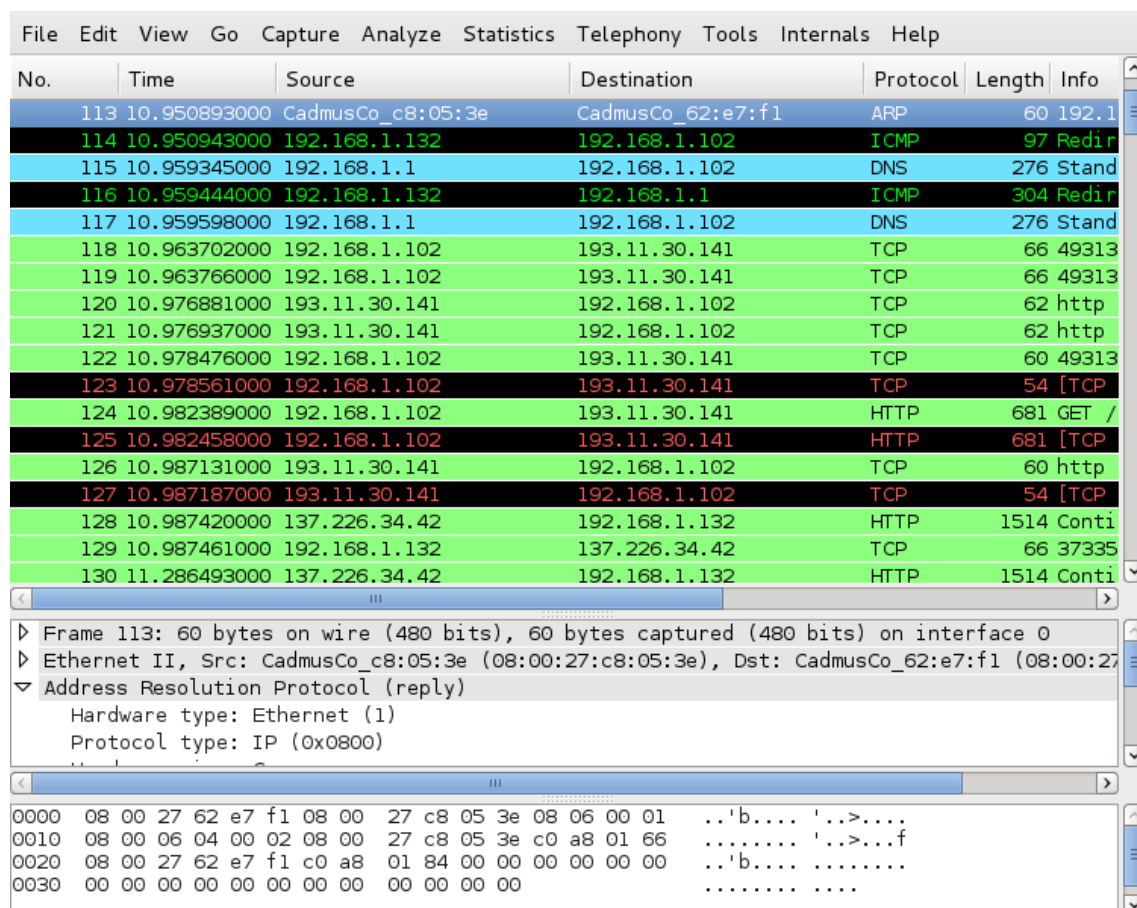
¹<https://www.archlinux.org/>

²<http://www.kali.org/>

4.1.3.1 Man in the Middle

To mount a Man in the Middle attack, the attacker connected to the same access point as the client was connected to. The attacker first used arpspoof³ to conduct an ARP Spoofing attack (see section 1.3.1 and 2.2.2.1) to position itself between the client and the access point. The attacker thus gained access to packets going in either direction.

After positioning itself between the client and the access point, the attacker used Wireshark⁴ to confirm that the attack was a success. The results from which can be seen in figure 4.1. The access point is at 192.168.1.1, the client at 192.168.1.102, and the attacker (the one capturing the traffic) is at 192.168.1.132. The client accessed <http://dsv.su.se/>.



The image shows a Wireshark packet capture interface. The top pane displays a list of captured packets with columns for No., Time, Source, Destination, Protocol, Length, and Info. The bottom pane shows the details of the selected packet (No. 113), including Ethernet II, Address Resolution Protocol (reply), and the raw packet data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
113	10.950893000	CadmusCo_c8:05:3e	CadmusCo_62:e7:f1	ARP	60	192.1
114	10.950943000	192.168.1.132	192.168.1.102	ICMP	97	Redir
115	10.959345000	192.168.1.1	192.168.1.102	DNS	276	Stand
116	10.959444000	192.168.1.132	192.168.1.1	ICMP	304	Redir
117	10.959598000	192.168.1.1	192.168.1.102	DNS	276	Stand
118	10.963702000	192.168.1.102	193.11.30.141	TCP	66	49313
119	10.963766000	192.168.1.102	193.11.30.141	TCP	66	49313
120	10.976881000	193.11.30.141	192.168.1.102	TCP	62	http
121	10.976937000	193.11.30.141	192.168.1.102	TCP	62	http
122	10.978476000	192.168.1.102	193.11.30.141	TCP	60	49313
123	10.978561000	192.168.1.102	193.11.30.141	TCP	54	[TCP
124	10.982389000	192.168.1.102	193.11.30.141	HTTP	681	GET /
125	10.982458000	192.168.1.102	193.11.30.141	HTTP	681	[TCP
126	10.987131000	193.11.30.141	192.168.1.102	TCP	60	http
127	10.987187000	193.11.30.141	192.168.1.102	TCP	54	[TCP
128	10.987420000	137.226.34.42	192.168.1.132	HTTP	1514	Conti
129	10.987461000	192.168.1.132	137.226.34.42	TCP	66	37335
130	11.286493000	137.226.34.42	192.168.1.132	HTTP	1514	Conti

Frame 113: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet II, Src: CadmusCo_c8:05:3e (08:00:27:c8:05:3e), Dst: CadmusCo_62:e7:f1 (08:00:27:c8:05:3e)
Address Resolution Protocol (reply)
Hardware type: Ethernet (1)
Protocol type: IP (0x0800)

0000 08 00 27 62 e7 f1 08 00 27 c8 05 3e 08 06 00 01 ..'b....'..>....
0010 08 00 06 04 00 02 08 00 27 c8 05 3e c0 a8 01 66'.>...f
0020 08 00 27 62 e7 f1 c0 a8 01 84 00 00 00 00 00 00 ..'b....
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00>

Figure 4.1: Results from analysing the traffic from the client with Wireshark, captured by the attacker.

4.1.3.2 Evil Twin

To set up an Evil Twin, the attacker used the aircrack-ng suit⁵. To broadcast the Evil Twin access point, the aircrack-ng tool in the suit was used to imitate one of the legitimate access points and airplay-ng was used to force the client to disconnect from the legitimate AP. As was argued in section 2.2.2.1, an attacker would probably use this type of approach. By forcibly disconnecting clients from the legitimate network, the attacker would also be

³<http://www.monkey.org/~dugsong/dsniff/>

⁴<http://www.wireshark.org/>

⁵<http://www.aircrack-ng.org/>

making sure that no client would connect to the legitimate access point and escape the attack. All clients that try to connect to the legitimate access points would be instantly disconnected and try to connect to the network again, which would result in eventually connecting to the attacker.

To affirm that the Evil Twin was indeed operational, Wireshark was used to monitor the traffic sent to and from the client. The results were similar as those seen in figure 4.1

4.2 Regarding chosen software

Section 4.1.3.1 and 4.1.3.2 mentions different software that will be used during the experiments. This section aims to explain why these were chosen.

4.2.1 Wireshark

Wireshark is a common network protocol analyser that is freely available from its website, <https://www.wireshark.org/>. It has become a de facto standard in many industries and educational institutions. Wireshark lets you monitor the activity in a network in real time and provides a helpful inspection into the packets that have been captured.

Due to its wide use, its big community, the ability to investigate packets in all their forms, and the fact that it is widely used in the network industry, Wireshark was chosen as the software used to monitor the network activity on the networks set up for the experiments.

4.2.2 arpspoof

Section 4.1.3.1 mentions arpspoof as the software to be used to set up the Man in the Middle attack. This is also a freely available software, found at <http://su2.info/doc/arpspoof.php>. The software is part of a bigger package, named *dsniff*. What arpspoof does is to send ARP packages to a user-defined address telling the target that the sender is the owner of an IP-address that the user has defined. While not backed by a big community, this software is among the most used tools for arpspoofing due to its simplicity. While there are plenty of alternatives, the simplicity and low resource usage of the software made it a prime choice according to the researchers.

4.2.3 Aircrack-ng

The aircrack-ng suit, mentioned in section 4.1.3.2, comes with a big variety of tools that can be used for wireless auditing. The software in the suite can be used for monitoring traffic, cracking WEP, WPA, and WPA2-PSK networks, sending fake packages to a client, perform Evil Twin Attacks, and much more. The suite is available at <http://www.aircrack-ng.org/> and is freely available.

Due to its versatility and the many tutorials found on the Internet where the suite is used, the researchers argue that an attacker would probably use this software as it is easy to get started with, due to its apparent wide popularity. The researchers therefore elected to use the aircrack-ng suite to perform the Evil Twin Attack.

4.3 Experimentation Set-up

This section will explain the network topology used in the experiments

4.3.1 Network Topology - Man in the Middle

In figure 4.2 the network topology used for the Man in the Middle attack is shown. The attacker is within radius of the same access points as the client is and both the attacker and the client can connect to either access point. As was described in section 4.1.1, the access points transmit a network named *RogueAPTesting* that requires no password or other authentication methods. This is to imitate the environment as can be found in coffee-shops or airports.

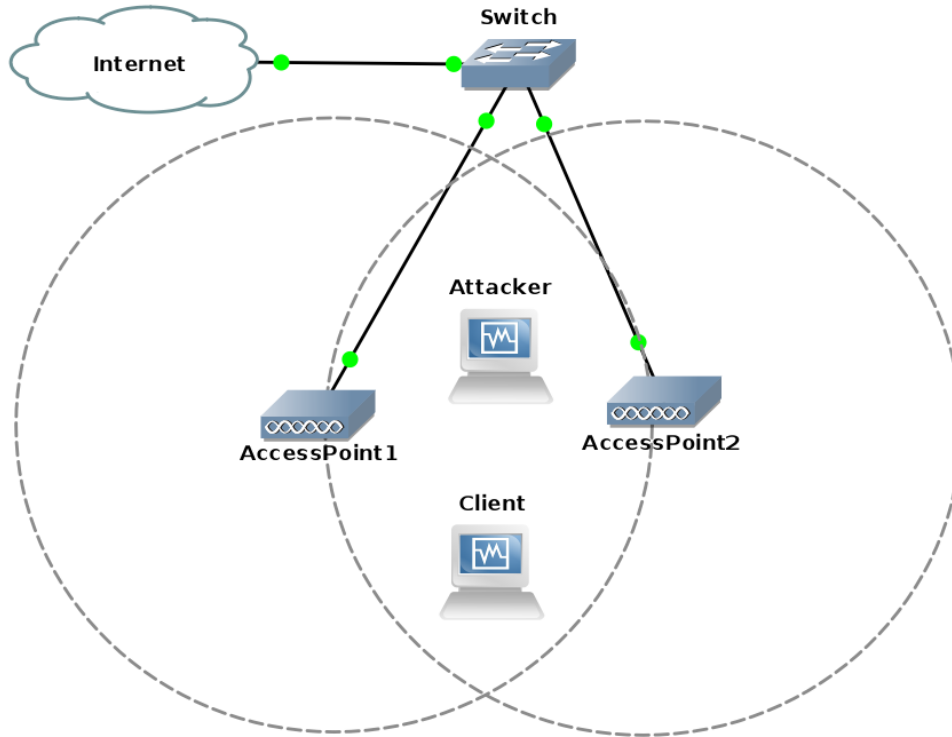


Figure 4.2: Network topology used in the experiment

4.3.2 Network Topology - Evil Twin

Figure 4.3 illustrates the network topology for the Evil Twin attack. The legitimate access points are still present, as they were in the network topology for the Man in the Middle attack. However, the attacker is now sending out its own signal which is a copy of one of the access points signals. The client will still only see two access points when in reality there are three devices broadcasting where one is used with malicious intent. The attacker hopes to overpower the legitimate access point, making the client connect to the Evil Twin instead of the legitimate access point.

4.4 Executing the Experiments

In this section, how the experiments were performed will be explained. As has been mentioned earlier in this paper, only one version of each attack will be performed. There are numerous ways to perform each of the attacks, but to test them all would require a study of its own. It was therefore decided that only one version of each of the types would be

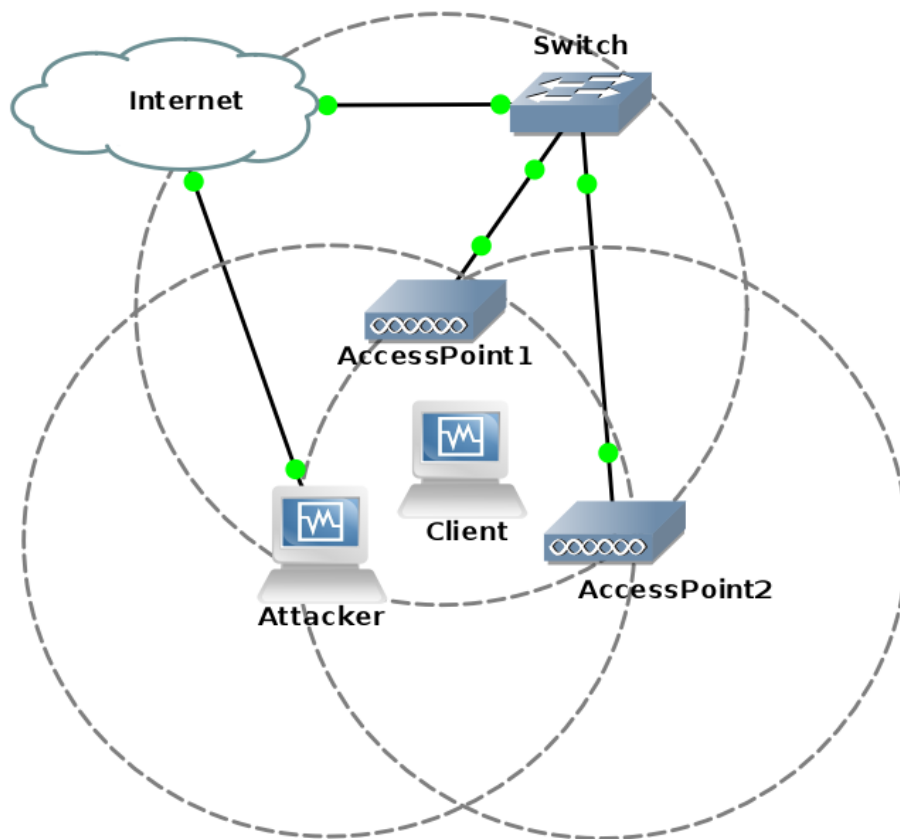


Figure 4.3: Network topology during the Evil Twin attack

tested. If the algorithm succeeds in testing these, a future study would be able to test other versions to see if these are also successfully detected.

4.4.1 Man in the Middle Attack

The algorithm proposed in Nikbakhsh et al. (2012) is supposed to be able to detect when a Man in the Middle attack is being performed. An experiment was therefore created to test this hypothesis. How this experiment was performed will be given as a list of steps with the actor of the steps noted at the beginning of the line. The list assumes that prior to the experiment, the access points have been configured as previously described and the client has connected to one of the access points.

1. [Attacker] Connect to an access point and ensure that the client is connected to the same one. A ping to the broadcast address should result in an answer from the client.
2. [Attacker] Perform an arp-spoofing attack against both the client and the access point.
 - (a) Run the command


```
arp spoof -i <NIC> -t <target_ip> <accesspoint_ip>
```

 where <NIC> is the network interface to use, <target_ip> is the target (the connecting client running CSWIDS), and <accesspoint_ip> is the access point.
 - (b) Open another terminal and run the command in reverse order


```
arp spoof -i <NIC> -t <accesspoint_ip> <target_ip>
```

3. [Attacker] Forward all packets with `ipforward`.
 - (a) In the terminal, run `echo 1 > /proc/sys/net/ipv4/ip_forward`
4. [Attacker] The MitM should now be in place. Verify with Wireshark that the attack was a success (see Figure 4.1).
5. [Client] Perform a task that will produce network traffic, eg. visit a website.
6. [Client] Run the CSWIDS program against both access points again and document the result.

4.4.2 Evil Twin Attack

The algorithm proposed in Nikbakhsh et al. (2012) should also be able to detect an Evil Twin attack. Described here are the steps performed to set up an Evil Twin to test the algorithm. The description will be given as it were in 4.4.1, with a list of steps and the assumption that the access points have been configured as described earlier and the client connected to one of them.

1. [Attacker] Connect to the Internet via a network card that is not the wireless one.
2. [Attacker] Put the wireless card in promiscuous mode.
 - (a) Run the command `airmon-ng start wlan0`
 - (b) Airmon-ng should now have renamed the network card to `mon0` when putting it in promiscuous mode.
3. [Attacker] Gather information about nearby networks using the newly created `mon0` from the previous step.
 - (a) Run `airodump-ng mon0`
 - (b) Information about nearby networks should now be presented in the terminal
4. [Attacker] Wait for the client to connect to an access point.
5. [Client] Connect to an access point.
6. [Attacker] Take the information found with `airodump-ng` and create the twin.
 - (a) Example information to use
 - MAC: 00:11:22:33:44:55:66
 - Network name: RogueAPTesting
 - Channel: 11
 - (b) Run `airbase-ng -a 11:22:33:44:55:66 \`
`--essid "RogueAPTesting" -c 11 mon0`
 - (c) The Evil Twin should now be broadcasting.
7. [Attacker] Deauthenticate the client from the legitimate access point.
 - (a) Run `aireplay-ng --deauth 0 -a 11:22:33:44:55:66`
 - (b) All clients connected to the access point should now have been disconnected.
8. [Client] Reconnect to the network.
9. [Attacker] If necessary, increase the power of the signal of the Evil Twin to overpower the access point.
 - (a) Run `iwconfig wlan0 txpower 27`
 - (b) The Evil Twin should now have a signal strength of 27 dBm or 500 milliwatts. This should be sufficient to overpower the legitimate access point.
 - (c) If 27 dBm was not sufficient to overpower the access point, increase the signal to 30 dBm, 1000 milliwatts.
 - i. Run `iw reg set B0`
 - ii. Run `iwconfig wlan0 txpower 30`
 - (d) Verify the output power with `iwconfig`

10. [Attacker] Verify with Wireshark that the Evil Twin is indeed operational and that the client has connected to the Twin. The resulting network topology should look as in figure 4.3.
11. [Client] Perform a task that will produce network traffic, eg. visit a website.
12. [Client] Run the CSWIDS program against both access points again and document the result.

4.4.3 Ethical Considerations

4.4.3.1 Network Name

Since the experiment would be able to gather any information that passed by the attacker, it was a possibility that a user who was not part of the experiment would connect to the open network that was set up. A name that would deter users from connecting to the network was chosen, *RogueAPTesting*. A user who sees the network would hopefully think twice before connecting to such a network. All the users in and around the laboratory where the experiment took place were told not to connect to "RogueAPTesting".

4.4.3.2 General Regarding Experiments

As described in Denscombe (2010:p. 329-342 (Appendix 1)), there are four key principles which underlie codes of research ethics. These are

1. Participants' interests should be protected
2. Participation should be voluntary and based on informed consent
3. Researchers should operate in an open and honest manner with respect to the investigation
4. Researchers should comply with the laws of the land

While this research is aimed at devices (objects) rather than people and has no participants other than the experimenters, the principles given are still important to follow. However, since there are no participants, the first two principles will not be in danger of being broken. The third and fourth principles are of more concern and will be given more thought.

Principle 3: Researchers should operate in an open and honest manner with respect to the investigation

This, according to Denscombe (2010:p. 335), has more to do with researchers being open about what they are doing when dealing with their participants. This research has no participants, but the experiment environment will be able to put others in "the line of fire" when doing the attacks. It is therefore important that the people near the experimentation site is informed about what is being done. A network name (*RogueAPTesting*) was also chosen for these experiments to raise warnings to people that has not been informed about the experiments.

Principle 4: Researchers should comply with the laws of the land

Although MitM and Evil Twin attacks are illegal in Sweden, the law only governs attacks where permission is not given to monitor the communication.(Justitiedepartementet L5, 2014) The experiments in this research was performed with permission and in a controlled environment. The law was therefore not broken and the experiment operated within the laws of the land.

5. Results & Analysis

In this chapter the results of the experiments as well as an analysis of the results.

It should be noted that the results presented display only the results from the experiments in this paper. They do not provide a generalised picture of how the algorithm performs with other setups. They do however give a hint of what the answer to other setups could look like, but these results are by no means the results of those.

5.1 Results

As the original algorithm was impossible to implement due to a flaw in the algorithm itself, the results presented are based on the algorithm presented in section 3.2.1.

The researchers performed 20 iterations of each experiment. It was deemed that a 5% representation of each iteration was enough to get a statistically valid result.

5.1.1 Man in the Middle

Out of 20 performed experiments, the algorithm detected 20 attacks.

5.1.2 Evil Twin

Out of 20 performed experiments, the algorithm detected 0 attacks.

5.2 Analysis

The analysis here uses the method described in section 2.3.

5.2.1 Man in the Middle

$$100\% = 1 = \frac{20}{20}$$

The algorithm successfully detected the Man in the Middle every time, giving it a 100% accuracy for the experiment.

5.2.2 Evil Twin

$$0\% = 0 = \frac{0}{20}$$

The algorithm was unable to detect the Evil Twin during any of the iterations of the experiment. The results are therefore a rock bottom 0% accuracy.

6. Conclusion & Discussion

6.1 Conclusion

The experiments performed in this study have shown that when mounting the attacks that were used in the experiments, the algorithm described in section 3.2.1 can detect Man in the Middle attacks that operates on layer 3 and decrements the Time To Live (TTL) of IP packets. The algorithm can however not detect Evil Twin attacks. This gives an overall success rate of 50% (100% of MitM + 0% ETA = 50% overall). The authors would however not recommend using the algorithm.

For further scrutiny of the algorithm, please see section 6.2

6.2 Discussion

While this study showed that the algorithm proposed in Nikbakhsh et al. (2012) could detect a Man in the Middle attack, the algorithm should not be used. Following will be a discussion of why it worked in this study, and why it will not work ‘in the wild’.

6.2.1 Why it worked

The reason the algorithm worked in one of the experiments in this paper was due to the nature of the attack. The algorithm detected the Man in the Middle attack by use of trace routes. Trace routes sends a packet with an TTL (Time To Live) as low as possible. The TTL defines how many ‘hops’ (how many devices are allowed to process the IP packet and decrement the TTL) is allowed for the IP packet sent. When the TTL reaches 0, the device should send back an ICMP packet with the message ‘Time Exceeded’, which means that the TTL is too low for the packet to be sent any further. Since every device that inspects the packet should decrement the TTL, the attacker in the study would do so as well. The attacker mounted an ARP spoofing attack and forwarded the packets sent to it. By default all packets TTLs will be decremented. The algorithm would therefore find the attacker as the attacker would send back an ICMP Time Exceeded response.

6.2.2 Why it would not work in the real world

Any attacker that would mount an attack with malicious intent would most likely know that doing so is illegal. The attacker would most likely also know that trace routes could show that this sort of attack is in progress. With this information, the attacker would probably take steps to avoid detection. In this case, it means not showing up in a trace route. In Linux, which is the operating system which carried out the attacks in this study, this can easily be done with the *iptables*¹ application. Iptables allows a user to specify how an IP packet should be handled. If the attacker were to use the flag `-ttl-inc 1`, any IP packet that matches the rule where this flag is used would not have a decreased TTL when leaving the attackers device. The reason to increase the TTL by one is because the code that handles network communication in Linux automatically decreases the TTL by one. By increasing it by one after receiving it, the TTL will look as unchanged, and the

¹<http://www.netfilter.org/projects/iptables/>

TTL would never reach zero and trigger a ICMP Time Exceeded response. The attacker would then become 'invisible' in the traceroute and would not be detected.

6.2.3 Flaws in the algorithm

This section will go through the algorithm proposed in Nikbakhsh et al. (2012) step by step and point out the flaws in it. If possible, a remedy will be suggested so the flaw can be avoided. If no remedy can be suggested, it means that the algorithm provides little room for improvement without redesigning it completely. Each step to be scrutinised will assume that the previous step was a success and is flawless.

6.2.3.1 Get IP from both APs (same SSID and BSSID)

As was discussed in section 3.2.1, this step is impossible to fulfil. If two access points send out the same information, a client will see them as one and the same access point. The client can therefore not test against two access points as only one can be seen.

In theory, this is a good way to detect an Evil Twin. The Evil Twin will send out the same information as the legitimate access point. It will therefore look as the same access point. Detecting two devices that are sending out the exact same information could therefore be a sign that an Evil Twin is present. The client however has small chances of success of seeing this, and to test them both would not be possible at present as they would register as the same access point.

Evil Twins can be detected with the use of Received Signal Strength (RSS) and frequency in radio transmission. A way to detect rogue access points using RSS was described in Kim et al. (2012). While this could detect that there is an Evil Twin present, the client still cannot connect to both as it will see them as the same device. The connection will be made to the device with the best signal strength.

The detection of the Evil Twin with RSS will still not amend the flaw in the algorithm proposed by Nikbakhsh et al. (2012). This is a flaw that the authors of this study cannot find a remedy for. The only solution found was to change the requirement and by that change further parts of the algorithm.

6.2.3.2 Are both IPs the same?

This is a dubious test. If the access points also act as a DHCP server, they might very well have the same IP and still be part of the same network. For example, if host A is hosting a DHCP server and hands out IP addresses to devices which connect to it, it might have the IP address of 192.168.0.1. Now suppose that host B and C connect to host A. B might get the IP of 192.168.0.2 and C could get 192.168.0.3. Now suppose that B and C also host DHCP servers of their own. Device D, that connects to B, might get the address 192.168.0.2 and will see B at address 192.168.0.1. Host C might be in a similar situation. B and C are part of the same network, but their IPs are the same as seen by the connecting client (see figure 6.1). According to the algorithm, this situation must mean that the access points cannot be trusted. In reality, this is a possible situation and does not mean that the network is unsafe. It is a matter of multiple levels of Network Address Translation (NAT).

"According to network logic, we could not have two same IP addresses in one network simultaneously. If this situation happens it will cause IP address conflict and both devices will stop working"

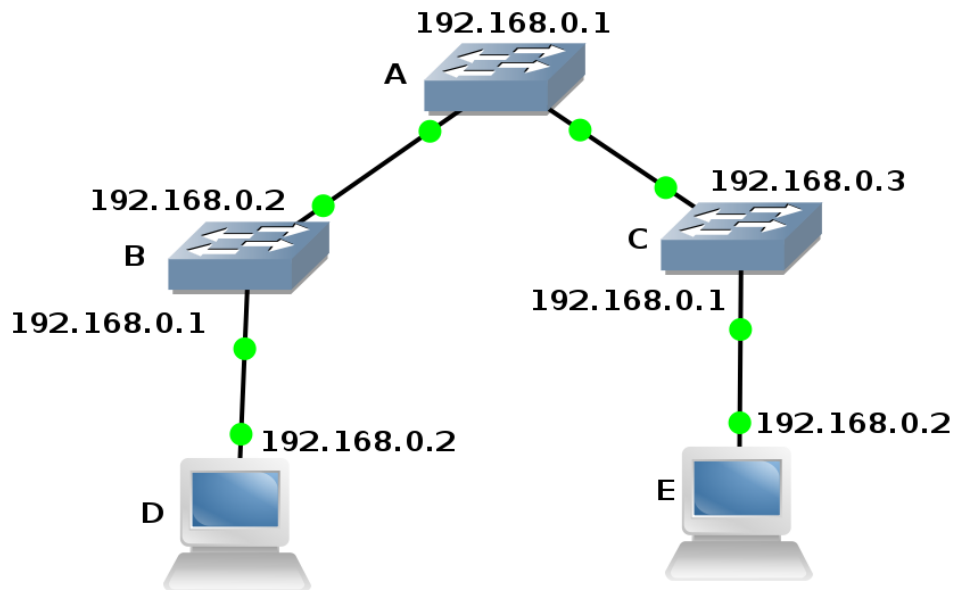


Figure 6.1: Multiple levels of NAT

(Nikbakhsh et al., 2012:p. 3)

This is not entirely true. Two devices could have the same IP and still operate in the same network. In fact, this is what happens when an attacker performs an ARP spoofing attack. Two devices will claim to have the same IP address, and both will still function. If this would actually cause the devices to stop working, ARP spoofing attacks would not be the problem they are today.

6.2.3.3 IPs are the same: trace route same destination from both APs

A trace route is a network diagnostics tool to help track the path a packet takes to reach a destination. The route that the packet travels along is determined at every node that handles the packet. The node looks at the destination for the packet and dictates to which next node it should send it so it can reach the destination as fast as possible. Due to a myriad of factors like usage, node-outage, adding and removal of nodes, network congestion, etc. the route that the packet travels does not necessarily need to be the same. The route is by no means a static route that will always be taken no matter what. This results in trace routes being an unreliable source of information to begin with. It can give information of how a packet travels, but the same path does not need to be taken by the next packet sent.

The use of trace routes in the algorithm is to detect the rogue access points. The hope is that the rogue will operate on layer 3 in the TCP/IP stack and decrement the TTL of the packet sent. This would make it show up in the trace route and be detected. As explained in section 6.2.2, this will most likely never happen as it is likely that the attacker will either operate on layer 2 or not decrement the TTL.

6.2.3.4 IPs are the same, trace route: Are the results the same?

By studying the algorithm and reading the arguments used in Nikbakhsh et al. (2012), the reader might wonder why this test is needed at all. The question is a simple boolean question, “*Are the trace routes the same?*”, but one of the outcomes is deemed impossible with the following reasoning

"[...]The first situation could not happen, because both access points have same SSIDs, MAC and IP addresses and packet travels exactly the same route. According to network logic, we could not have two same IP addresses in one network simultaneously. If this situation happens it will cause IP address conflict and both devices will stop working. Therefore, in Figure 2 this situation is shown as "N/A". Therefore the only answer would be the same IP addresses with different trace routes."

(Nikbakhsh et al., 2012:p. 3)

The reason to conduct the trace route in the first place is therefore also not needed, as there could only be one answer (the trace routes differ) to whether the networks tested are safe or not if they share the same IP. If the authors provide a test, and at the same time dictate that one of the two possible outcomes are impossible (N/A), then the only possible outcome of the test is the other option, so why conduct the test in the first place?

The assumption that there can only be one possible outcome is however wrong. It is based on the argument that 'network logic' would prevent two hosts to have the same IP address. As discussed above in section 6.2.3.2, this is a argument without basis. An Evil Twin would for example not show up on such a trace route as different. Two access points of the same model set up under the same situation as described in 6.2.3.2 would not show any differences if the trace route actually took the same route after leaving the network. The trace route would look exactly the same, and the claimed impossible outcome is, in fact, possible.

6.2.3.5 IPs are different: Net ID is the same?

"In the first step, which is IP comparison another result is different IPs. In order to make decisions, this method calculates network IDs, based on different IP classes and compares them. If network IDs are the same, it indicates that both APs are in the same network. This situation is the result of load balancing in the network."

(Nikbakhsh et al., 2012:p. 3-4)

The network ID is the network part of the IP address. The IP address 192.168.0.1/24 is defined in CIDR² notation. The first 24 bits of the address is the network identifier (and has also become to be the subnet mask)(Microsoft, 2008). The 8 bits that are left identifies the host address. In this example, the Net ID would be 192.168.0. All devices that belong to the same network should have an IP that begins with this. If a device does not have this as the leading octets, it is not part of the network.

Suppose that two access points, A and B, are sending out the same network name. Suppose access point A has an IP of 192.168.0.1/24 and B has an IP of 192.168.0.100/24. Both networks have a Net ID of 192.168.0 and therefore, according to the algorithm, they belong to the same network and are both marked as 'Safe'. But what if access point B is a rogue access point and connects to an entirely different network? Then both will still be marked as 'Safe'.

Clearly, this is wrong. The rogue is marked as safe when it has the same network name, BSSID, and NetID as the legitimate access point. This can easily be forged and cannot be trusted as way to identify rogues.

The other possible outcome of the NetID test is that they are not the same. If this is the case, it means that the access points tested is simply not in the same subnet. Access point A might have an IP of 192.168.0.1 and access point B might have an IP of 192.168.1.1. These are now in a position where the algorithm have deemed them unsafe, even though this is a possible scenario for legitimate use. Consider a network

²http://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing

where the administrator wishes to separate device IP addresses based on where they are in the building. The clients would all still connect to the same network, but the administrator can see, based on the IP of the client, where in the building the device is located. The algorithm would see these access points as unsafe.

6.2.3.6 NetID not the same: Trace route same destination from both APs

At this point, the algorithm performs a trace route to determine if there is a Man in the Middle present on any of the access points. If one of the trace routes is longer than the other, the algorithm claims the presence of a Man in the Middle. As was discussed in 6.2.2 and 6.2.3.3, trace routes are not a safe tool to detect a Man in the Middle. Further explanation of why trace routes should not be used will not be discussed.

6.2.3.7 Is there an extra hop in trace route?

As has already been discussed, this is not a safe way to determine the presence of a Man in the Middle. The attacker can easily hide its presence, and the trace route could show false positives as well as false negatives (an attacker is present, but the trace route takes a faster route). This is of course unacceptable since the act of hiding ones presence is done with such ease, making the trace route as good as useless to detect anything but the most basic attacks.

No matter the outcome of the test, the result is that the user should probably not connect to the access point. One cannot be sure that there is an attack in progress, but it is suspected. The other outcome is a definitive (according to the algorithm) proof that there is an attack in progress.

6.2.4 The Algorithm, Simplified

The algorithm proposed in Nikbakhsh et al. (2012) makes assumptions that are not rightly justified. The outcomes of the algorithm, and the entire algorithm, can be summarised in a shorter and more simple algorithm. This is shown below in figure 6.2. The flowchart tries to illuminate the errors of the original algorithm.

The errors being highlighted are

1. Trying to test two access points with the same SSID and BSSID
2. Deciding, based on IP, whether the access point is safe or not.
3. Deciding, based on NetID, whether the access point is connected to the same network or not.

These are the main three flaws in the algorithm that can be found. The other flaw is using trace route to try and detect an attacker. While a simple trace route could detect an extra hop, this is not a sure sign that an attacker is present or not.

6.2.5 Ethical Implications

The ethical implications of this research are limited. No new attack vector has been exposed, and no new way of detecting rogues has been found. The only possible new information that have been revealed is how useful (or perhaps, how not so) the algorithm proposed in Nikbakhsh et al. (2012) actually is.

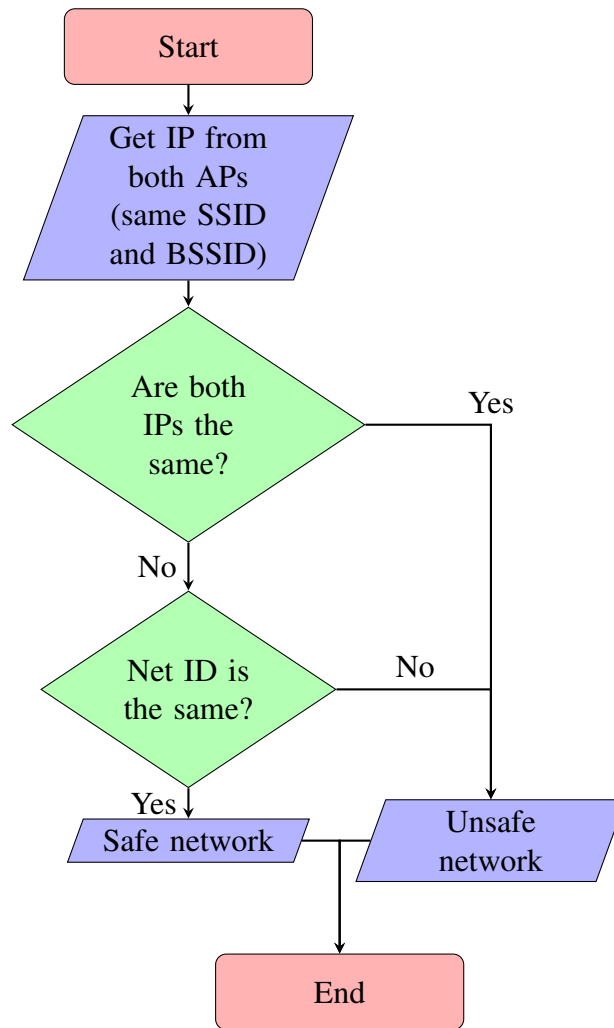


Figure 6.2: Simplified version of the algorithm proposed in Nikbakhsh et al. (2012)

6.2.6 Limitations of the Study

This study set out with the goal to investigate how well the algorithm proposed in Nikbakhsh et al. (2012) would perform in detecting rogue access points that implemented Man in the Middle and Evil Twin attacks. As there are several ways to perform these attacks the researchers of this study chose to use one method to perform the MitM, and one method to perform the ETA. This was to keep the study from becoming too big in regards to all ways an attacker can perform the attacks as this was not the aim of the study. The aim was to test the algorithm, not to test all ways to perform these types of attacks. This study is thus limited to using only one method respectively for each type of attack.

A study that uses different methods for the attacks might get different results than this study. The researchers of this study are however doubtful that it would.

A difference that would affect the value of the algorithm negatively, which has been discussed earlier, is in the case of the Man in the Middle attack. If the attacker does not decrement the TTL, the attacker would not show up in the trace routes. The algorithm would therefore not detect the MitM. The researchers of this study have not been able to think of any other, realistic, ways that would affect the algorithm.

Due to the limitation of this study regarding the choice to limit the research to only one attack method of each attack type, the results should not be seen as a generalisation of how the results of all attack methods will be. This research presented one attack method for each attack type. The results therefore only represent the results for these attack methods. The discussion that followed the results should however be seen as a general guide line. The discussion breaks down each step in the algorithm and tries to explain why this is not something that should be seen as a sound algorithm. There are numerous pitfalls that should be avoided when creating an algorithm that should detect rogue access points. The proposed algorithm falls into several of them, which the discussion has shown.

If the experiments would have been conducted in a different manner, as has been suggested previously in section 6.2.2, the results would be different. Yet, the changes that can be made, as seen by the researchers, would only serve to diminish the algorithms likelihood of detecting the attacker. As could be seen in the experiment that tested for an Evil Twin, the algorithm was unable to detect the attacker even once. This is because the algorithm relies on traceroutes, which as has been discussed, are not a reliable source of information. The researchers of this study therefore argues that the results from these experiments are the best results that the algorithm will ever achieve. The results should therefore be seen as a cap of how well the algorithm can perform, not a generalisation of how it will perform with other experimentation setups.

6.2.7 Future Research

Future research into the algorithm proposed in Nikbakhsh et al. (2012) will not be encouraged by this study. The algorithm has proved to fail on most accounts and future research should focus their aim at other algorithms and methods to detect rogue access points.

Roth et al. (2008), Aziz and Hamilton (2009), and Han et al. (2009) all provide different methods for detecting rogue access points from the client side.

Roth et al. (2008) describes a way that will help detect an Evil Twin Attack with the help of SAS. This forms a good basis for securing the communication to and from the client and the gateway. By first establishing that the access point is the legitimate access point with the help of key phrases (or passwords) the risk of connecting to a rogue is reduced. This, in contrast with relying on traceroutes, is a much more secure way to establish reliability. While the attacker might learn what phrases should be used, the attacker must actively search for these. The method also does not suffer from having to try and detect two devices that is sending out the same information.

Aziz and Hamilton (2009) and Han et al. (2009) both describe a method where timing between sending a message and receiving a message would differ depending on someone intercepting it or not. This again provides more reliable information than relying on traceroutes and looking for radio waves coming from two different sources. While the timing might give false positives due to network lag and other such circumstances, it has the possibility to take this into account and make adjustments for it, giving a better accuracy.

While these do not promise to detect both ETAs and MitMs, they provide the ability to detect rogues. If one of the algorithms provide the ability to safely detect a rogue, it could, perhaps, with further study or by using several different methods be able to also detect what kind of rogue it is. The researchers of this study encourages research into these methods rather than the one provided in Nikbakhsh et al. (2012).

Bibliography

- ABI Research (2013). Growing demand for mobility will boost global wi-fi hotspots to reach 6.3 million in 2013, <https://www.abiresearch.com/press/growing-demand-for-mobility-will-boost-global-wi-f>.
- Aziz, B. and Hamilton, G. (2009). Detecting man-in-the-middle attacks by precise timing, *Emerging Security Information, Systems and Technologies, 2009. SECURWARE '09. Third International Conference on*, pp. 81–86.
- Bauer, K., Gonzales, H. and McCoy, D. (2008). Mitigating evil twin attacks in 802.11, *Performance, Computing and Communications Conference, 2008. IPCCC 2008. IEEE International*, pp. 513–516.
- Beyah, R., Kangude, S., Yu, G., Strickland, B. and Copeland, J. (2004). Rogue access point detection using temporal traffic characteristics, *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, Vol. 4, pp. 2271–2275 Vol.4.
- Brain, M., Wilson, T. V. and Johnson, B. (2001). How wifi works, <http://computer.howstuffworks.com/wireless-network.htm>. Last accessed 2014-12-03.
- Corazza, G. C. (1998). Marconi's history, *Proceedings of the IEEE*, VOL. 86, NO. 7, pp. 1307–1311. <http://courses.washington.edu/ee420/handouts/marconi.pdf>.
- Denscombe, M. (2010). *The Good Research Guide*, McGraw-Hill International.
- Dhamija, R., Tygar, J. D. and Hearst, M. (2006). Why phishing works, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, ACM, New York, NY, USA, pp. 581–590.
URL: <http://doi.acm.org.ezp.sub.su.se/10.1145/1124772.1124861>
- Gast, M. S. (2005). *802.11 Wireless Networks: The Definitive Guide, Second Edition*, O'Reilly Media, Inc.
- Geier, J. (2003). Identifying rogue access points, <http://www.wi-fiplanet.com/tutorials/article.php/1564431> **4**.
- Godber, A. and Dasgupta, P. (2003). Countering rogues in wireless networks, *Parallel Processing Workshops, 2003. Proceedings. 2003 International Conference on*, pp. 425–431.
- Han, H., Sheng, B., Tan, C., Li, Q. and Lu, S. (2009). A measurement based rogue ap detection scheme, *INFOCOM 2009, IEEE*, pp. 1593–1601.
- IEEE (2012). Part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications, *Technical report*, IEEE Computer Society, 3rd Park Avenue, New York, NY 10016-5997, USA. <http://standards.ieee.org/getieee802/download/802.11-2012.pdf>.
- Johannesson, P. and Perjons, E. (2012). *A Design Science Primer*, first edn, CreateSpace.

- Justitiedepartementet L5 (2014). Brottsbalk (1962:700). Kap. 4 §9c.
- Kaufman, C., Perlman, R. and Speciner, M. (2002). *Network Security: Private Communication in a Public World, Second Edition*, second edn, Prentice Hall Press, Upper Saddle River, NJ, USA.
- Kim, T., Park, H., Jung, H. and Lee, H. (2012). Online detection of fake access points using received signal strengths, *Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th*, pp. 1–5.
- Kurose, J. F. and Ross, K. W. (2012). *Computer Networking: A Top-Down Approach*, sixth edn, Pearson.
- Lemstra, Wolter, V. H. and Groenewegen., J. (2010). *The Innovation Journey of Wi-Fi: The Road to Global Success*, Cambridge University Press.
URL: <http://dx.doi.org/10.1017/CBO9780511666995>
- Lockhart, A. (2006). *Network security hacks*, O'Reilly Media, Inc.
- Ma, L., Teymorian, A. Y., Cheng, X. and Song, M. (2007). Rap: Protecting commodity wi-fi networks from rogue access points, *The Fourth International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness & Workshops, QSHINE '07*, ACM, New York, NY, USA, pp. 21:1–21:7.
URL: <http://doi.acm.org/10.1145/1577222.1577252>
- Microsoft (2008). Ipv4. [Electronic] Accessed on 14-04-2014.
URL: <http://technet.microsoft.com/en-us/library/cc754783%28v=ws.10%29.aspx>
- Nikbakhsh, S., Manaf, A., Zamani, M. and Janbeglou, M. (2012). Advanced information networking and applications workshops (waina), 2012 26th international conference on, *A Novel Approach for Rogue Access Point Detection on the Client-Side*, pp. 684–687. <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6185342>.
- Roth, V., Polak, W., Rieffel, E. and Turner, T. (2008). Simple and effective defense against evil twin access points, *Proceedings of the First ACM Conference on Wireless Network Security, WiSec '08*, ACM, New York, NY, USA, pp. 220–235.
URL: <http://doi.acm.org/10.1145/1352533.1352569>
- Shetty, S., Song, M. and Ma, L. (2007). Rogue access point detection by analyzing network traffic characteristics, *Military Communications Conference, 2007. MILCOM 2007. IEEE*, pp. 1–7.
- Song, Y., Yang, C. and Gu, G. (2010). Who is peeping at your passwords at starbucks? - to catch an evil twin access point, *Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on*, pp. 323–332.
- Tanase, M. (2003). Ip spoofing: an introduction, *Security Focus* **11**.
- Wei, W., Suh, K., Wang, B., Gu, Y., Kurose, J. and Towsley, D. (2007). Passive online rogue access point detection using sequential hypothesis testing with tcp ack-pairs, *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, IMC '07*, ACM, New York, NY, USA, pp. 365–378.
URL: <http://doi.acm.org.ezp.sub.su.se/10.1145/1298306.1298357>

Yeo, J., Youssef, M. and Agrawala, A. (2004). A framework for wireless lan monitoring and its applications, *Proceedings of the 3rd ACM Workshop on Wireless Security, WiSe '04*, ACM, New York, NY, USA, pp. 70–79.
URL: <http://doi.acm.org/10.1145/1023646.1023660>

Stockholm University
Department of Computer and Systems Sciences
Forum 100
SE-164 40 Kista
Phone: +46 8 16 20 00
www.dsv.su.se



Stockholm
University