Detection of Compromised MACs within an Intranet

Arjun Sethi
The University of Texas at Dallas
Richardson, Texas, USA

Khoa Nguyen
The University of Texas at Dallas
Richardson, Texas, USA

ABSTRACT

This paper is a brief talk about internet router security and creating network attacks by spoofing Media Access Control (MAC) address. This address is a hard-coded information on a Network Interface Card (NIC). In a network, the MAC address is used in the ARP cache to establish the identity of a computer. However, using a Linux computer, we can use the drivers and tools on the Linux OS to change the MAC address thus changing the identity of a computer.

We want to explore this method to simulate an ARP spoofing attack by changing the MAC address on a Linux computer and tell other computers in the same network that the current computer is indeed another computer. The experiments were divided into three parts. In the first part the attacker assigns itself a client's MAC address. In the second part the attacker assigns itself a client's IP address and only sends the update to the router. In the third part the attacker assigns itself the router's MAC address.

The experiments utilized client server script written in Node.JS to simulate messaging between clients, the attacker, and the router. The first experiment was the most successful out of the three in creating a denial of service and data stealing attacks. The second experiment allowed successful man in the middle attack, however, it is fairly simple for the communicating parties to figure out who the attacker is. The last case could have been successful given the correct resources, but still provided excellent insight on router security. In conclusion, the experiments all found that router systems do not have reliable security protocols in place once a user knows the password to connect to it, and much of the security is dependent upon the operating systems of network connected clients.

CCS CONCEPTS

• **Security and privacy** → **Intrusion/anomaly detection and malware mitigation**; **Intrusion detection systems**; • **Software and its engineering** → *Software organization and properties*; • **Computer systems organization** → Architectures;

KEYWORDS

MAC,spoofing, compromise, IP security, router security, Hardware level security

ACM Reference Format:
Arjun Sethi and Khoa Nguyen. 2019. Detection of Compromised MACs within an Intranet. Richardson, TX, USA, 7 pages.

1 INTRODUCTION

When a device connects to a network, the device is assigned an IP address based on its Media Access Control Unit (MAC) address, which is the first unique identifier utilized to configure devices connecting to a network.[3] The router maintains an Address Resolution Protocol (ARP) table that pairs IP addresses and MAC addresses together. Networks rely on the assumption that MAC addresses cannot be the same for devices in a network. Unfortunately, this is a terrible assumption because manufacturers allow MAC address changes quiet easily. [8] The purpose of the experiments in this paper is to find out what happens if device MAC addresses are the same within a network, device IP addresses are the same within a network, and if device MAC and/or IP address is the same as the gateway to the internet. If a successful technique is found to confuse a network topology of who has what MAC and/or IP address, we investigate different network attacks that can be created and whether or not the attacker can be easily identified.

The first problem investigated is what happens when two devices within a network have the same MAC address. The dynamic host configuration protocol (DHCP) is a protocol that manages IP address to MAC address allocation. [7] Allocation is primarily dependent upon a client's unique ID, or more popularly known as a MAC address in networking. Routers utilize the DHCP protocol to assign an IP address to a device that connects to it. The IP address is utilized by the client to communicate with other devices over the network. For example, when Alice, Bob, and Comet connect to a network, each user will be assigned their own IP address if they have differing MAC addresses. This proposes a potential issue within modern day routers if they are not equipped to handle a case where Alice and Comet have the same MAC address, i.e. they no longer have a unique id. [5]

The second problem investigated is the DHCP protocol can be bypassed by assigning a static IP address to a device when connecting to a network. Instead of dynamically assigning an IP address to a MAC address, a device can command the router to store a certain IP address with its MAC address in the ARP table. When Alice connects to a router, the router will dynamically assign an IP address to her device. Before Bob connects to a router, he can assign his device the IP address that Alice is currently utilizing and then connect to the network. Most operating systems make it easy to claim an IP address to their device, which can potentially cause issues within the routing table of a network. The idea here is to see what happens to Alice, and if Alice can even tell quickly that her IP address was stolen by Bob.

The third problem investigated is that a device connected to a network can change its MAC address to the router's MAC address. The router is the gateway to the internet, so if the router cannot even be uniquely identified within a network, everyone within that network will be under attack.

2 ASSUMPTIONS

The experiments defined and results found in this paper were completed based on the assumption that all users including the attacker

are already within a network. To connect to a network, a user needs to know the correct credentials, such as WiFi password. The attacker in this paper could potentially be someone who has obtained credentials and is unauthorized to be inside of a network, or they can be someone who is authorized to access the network but violates trust within the system. It is a common case for people who are believed to be innocent actually have an underlying motivation to perform network attacks, such as within a work place where an malicious employee attempts to hack their company's system to steal information, or even possibly pose as another employee to cause harm.

3 EXPERIMENTATION

3.1 Technology Utilized in Testing

A Linux environment is used as the preferred operating system for the attacker in this case as Linux command line utilities provide more control over a computer's configuration settings, and the connected network configuration. There are many penetration testing tools available for anyone to use. There are several ways to change the MAC address on Linux, and specifically `ifconfig` is utilized in this experiment. `ifconfig` is an network-interface configuration utility.[2] When changing a network interface's MAC address, the device must be restarted for the changes to take effect. [6] For this reason, the network interface must be disabled, the MAC address changed, and then enabled as follows:

```
sudo ifconfig <interface> down
sudo ifconfig <interface> hw ether <new MAC>
sudo ifconfig <interface> up
```

For testing purposes, simple scripts written in Node.js are utilized to send messages between clients. This made it easier to see if an attack is successful or not. Computer A was utilized to send a message, Computer B would and C are utilized to receive messages. If B received the message, the attack is unsuccessful. If C received the message irrespective of B receiving the message, the attack is considered successful.

The sending message scrIPt was as follows:

```
var dgram = require('dgram');
var msg = Buffer.from('Hello.I am Alice!');
var client = dgram.createSocket('udp4');
client.send(msg, 3000, <Bob IP Address>);
```

The receiving message scrIPt was as follows:

```
const dgram = require('dgram');
const receiver= dgram.createSocket('udp4');
receiver.on('message', (msg, rinfo) => {
  console.log( Message from Alice: ${msg} );
});
receiver.bind(3000,<Bob IP Address>);
```

3.2 Spoofing MAC Address

The goal of this experiment is to find out if an attacker can make their device connect to a network and impersonate another user without being noticed. The MAC address is the first device identifier used in a network for IP assignment. If a new MAC address is detected, a new IP is assigned to the device. If an old MAC address is detected, the IP address is looked up in the ARP table, and the router will give the attacker the same IP address utilized in the previous network session. If an attacker is capable of making them self look like another user within a network, anyone who attempts to communicate with the user whose identity is stolen will not know they are communicating with an attacker.[5]

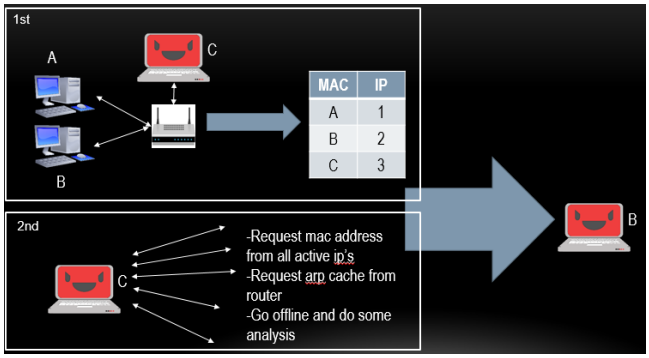

Figure 1: Attacker looks like someone else within a network.

Consider a case where Alice, Bob, and Comet are all connected to a network. Comet is a malicious user and would like to see what Alice is saying to Bob, and the best way to do this is if Comet looks like Bob. If Comet looks like Bob, Comet can request information from Alice. Alice, thinking Comet is Bob, can unknowingly send secret information to Comet.

Considering the MAC address is supposed to be a unique identifier within a network, a simple technique Comet can do is take Bob's MAC address and then reconnect to the router. The router will then think that Bob's device is Comet's device. When Alice sends a message to Bob's IP address, it will go through the router, the router will look up the IP address in its table and find Bob's MAC address, and finally the router will send the message to Bob's device. Since Comet has Bob's MAC address within the network, the message will actually be received by Comet's device.

Figure 1 above depicts the attack in this experiment. First, Alice, Bob, and Comet will connect to the network router, and the router will dynamically assign each of them an IP address. The MAC and IP are saved in the ARP cache table. Second, the attacker can simply ask Alice and Bob for their MAC addresses or request the router for the stored ARP table. Comet can figure out that Bob's MAC address, go offline, assign its device Bob's MAC address, and finally when the device reconnects to the system Comet will look like Bob. Comet's table entry will show that his device is offline.

(1) Computers A,B,C connect to network
(2) A sends message to B, B receives message. Both users think they are communicating with each other.
(3) C assigns itself B MAC address and reconnects to network
(4) Router finds B MAC address is in ARP table and B is still connected
(5) Router gets confused, and disconnects real B from network and gives connection to C.
(6) Router assigns C, B's IP address that was associated with real B's MAC

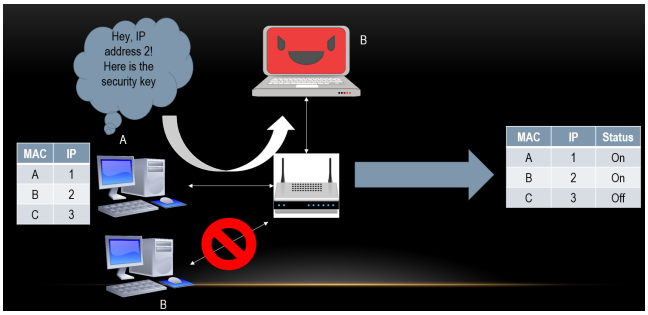

Figure 2: Attacker obtains MAC of another device. C is considered offline, the real B is kicked off, and C now looks like B to the network.

(7) C sends a gratuitous ARP to A to let him think that B is still online. Since C MAC address is the same as B, A does not make any updates to its ARP table and does not realize that C has B's identity
(8) A sends message to B, but C receives it

Figure 2 shows a diagram of the process detailed above.

3.2.1 Security Concern. Figure 2 shows that if C assigns itself B's MAC address and then reconnects to the network, a denial of service attack will occur on the real B's device. The router will take B's network connection and give it to C since C is a newer connection that has the same MAC address as the real B. Although C's MAC and assigned IP will remain in the router's ARP cache, there is no device connected to the network with C's real MAC address and thus C will appear as offline to the network. The associated IP with B's MAC address will be assigned to C since it appears as B.

Aside from denial of service to the real B, if A sends a message to B's IP address that it collected into its local ARP table the message will be routed correctly according to the router. However, it will be going to the C not the real B. This is a major concern considering A could have sent sensitive information that only B should have received. Furthermore, B cannot even let A know that it was kicked off the network because it has no access to the network.

We noticed that if B is connected to the network and C attempts to connect to the network using B's MAC address, B will get kicked off. If B attempts to reconnect, this will kick C off of the network similar to step 5. If B gets kicked off, then this is good because C is able to talk to A without A knowing, and B isn't able to tell A that it is kicked off the network because it can't even connect to the network unless it changes its MAC address. C can collect information destined for B, and the network is incapable of figuring out that C is acting as B since his unique identifier has been changed. If C gets kicked off, then this is not good because our attack is unsuccessful; C can also run a script to continuously reconnect to the network if it is kicked off. This would continuously provide a denial of service to B every time it attempts to connect to the network.

We also noticed that if B is disconnected to the network, the ARP cache tables are not flushed quick enough. C is able to connect to the network using B's MAC address and the network will think B is connected to the network. The router will not be confused because there are no longer two devices on the network that have the same MAC address at the same time so no device will be under a denial of service attack. C can collect information from A, leave the network, and the ARP table in the router will list both B and C offline. The next time the real B connects to the network, the device will not be notified of the imposter that acted as it while it was gone because the network itself was unable to detect the identity theft. At least when B is connected to the network, it could potentially be able to tell that it is being attacked, however, if it is not connected and a device temporarily utilizes its identification no trace is left and no one will know that it was an attacker posing as B for some time. Furthermore, if A had sent a message to B while B was offline and C was using B's MAC address in the network, C will receive the message and can leave the network without leaving a trace.

3.3 IP Address Spoof to Obtain MAC Spoof

The goal of this this experiment is to steal another devices IP address within the network to steal information from a communicating party. The purpose of this section is not to act a man in the middle as typical IP spoof attacks are, but rather an attack where the IP address is stolen to give the image of the attacker as the other end of a communicating party.

The process of stealing an IP address of a user on the network is as follows:

(1) Computers A,B,C connect to network
(2) A sends message to B, B receives message. They continue to communicate with each other
(3) C takes B's IP address by assigning a static IP address to it self
(4) C sends router gratious ARP that he has a new IP address, and the router updates its ARP cache table
(5) C sends A gratious ARP that he has a new IP address, and A updates its ARP cache table
(6) B gets a new IP address assigned to it
(7) A sends message to B's old IP address, but C receives it

Figure 3 defines an instance where everyone (A,B,C) connect to the network normally. The purpose of this is so that all ARP tables can be filled in, especially B's ARP table.

3.3.1 Security Concern. A user should not be able to hijack an IP address when connecting to a system! Instead of the router getting confused, an error should be thrown at the user attempting to use a static IP address that it is already in use by another machine on the network. Applications speak to each other normally by specifying IP address, thus if an IP address changes and the application is not notified, what if the message is sent to someone without the application knowing? This is exact what occured in this approach. Recall the two scripts utilized to send messages from A to B. In that script, a simple message was sent to B's IP address, so once C stole B's IP address the application continuously sent messages to the same IP address. However, C began to receive the messages not B.

Of course, this approach is easily detectable by building in the script to check if the assigned IP has a MAC address change. However, the purpose of this approach was to show that the security concern comes with stealing an IP address that is already in use. Router's should only allow IP address stealing for reserved IP and

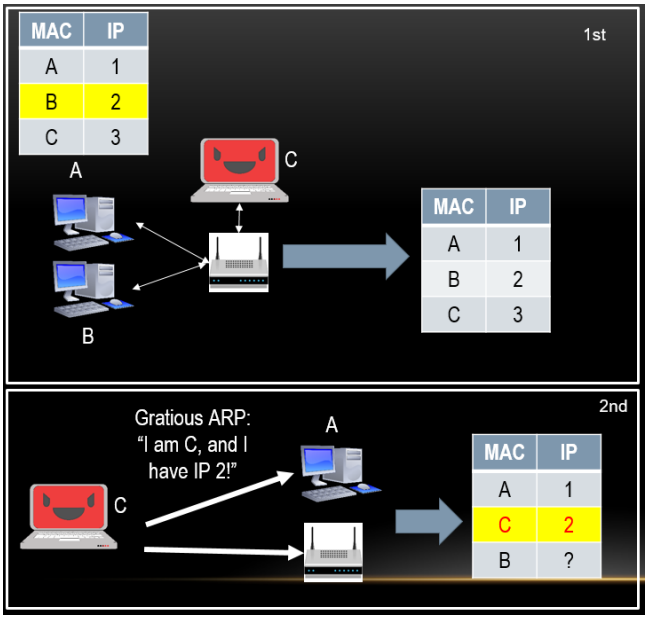

Figure 3: 1st all devices connect and ARP tables are created, then C sends an ARP to only router and A and only their tables are updated.

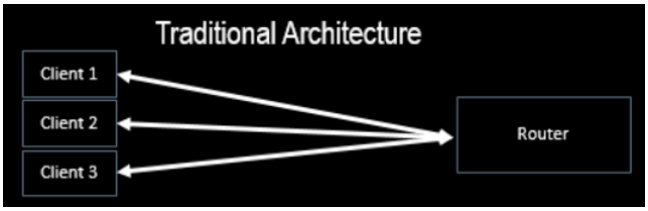

Figure 4: Normal architecture of devices connecting to the internet.

MAC associations within its configuration, otherwise deny any new requests to that IP if it is already in use.

3.4 Router Spoof

The goal of this experiment is to have a device that is normally supposed to use the router as a gateway to the internet act like the router by taking its IP and MAC address. Traditionally, all clients connect to the internet as shown in figure 4. If an attacker machine makes itself look like the router, it should still provide results as expects to the clients in the network. In other words, a user should be able to connect to the router like normal and access the internet as they normally would. This means that the attacker would have to become form a middle man attack between clients and the router. Any packets sent or received should just pass through the attacker, and then be forwarded to either the router or the client depending on the packet. This type of network architecture is diagrammed in figure 5.

The first step in creating this type of attack was to verify that on common operating systems, the attacker appears as the router

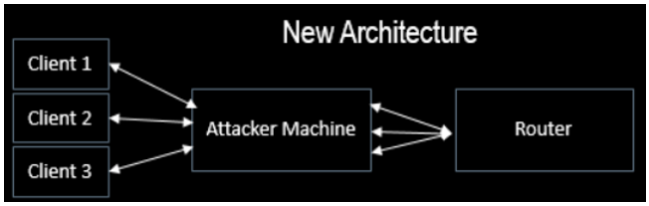

Figure 5: Man in the Middle attack with the attacker showing as the router of the network in the client's viewpoint.

in the available network connections panel. In this approach, Windows machines were utilized as the victim machines. The attacker machine was also running a Windows operating system because they have a built in hotspot feature. The hotspot feature allows network devices to share an internet connection through the windows machine.

The following steps were taken to assign the router's MAC address to the attacker:

(1) Connect to router and obtain its MAC address
(2) Assign MAC address to attacker machine
(3) Connect to router again
(4) set SSID and password of the hotspot to the same as the router's SSID and password
(5) Turn on hotspot feature and broadcast a WiFi connection that looks like the router's broadcast

After the hotspot is turned on and the attacker machine appeared as the router, simply waiting until a machine connects to the router would deem a successful attack if the connection went through the attacker's machine. Clients should be able to connect to the internet as they normally would:

- If a new machine wanted to connect to the router and have the correct password, it should be able to do so without error.
- If a device that has the router configurations saved from a previous connection, it should be able to automatically connect without error.

The steps mentioned in this attack will set up an architecture similar to the one referenced in figure 5, however, the client should have no reason to think that they are connected on a network with an architecture similar to the one referenced in figure 4.

3.4.1 Security Concern. The first result desired was for only one SSID to be broadcast to other devices, and this was successful. In the experiment, the SSID of the router was named "NETGEAR", had a password of "123456789", and a MAC address of "00-1f-33-f7-62-87". Figure 6 shows the ARP table locally stored on a victim machine as well as the available wifi connections. Figure 7 shows the ARP table update after the hotspot on the attacker machine has been turned on. Notice that even though one would expect both "NETGEAR" devices to display in the WiFi connection panel, only one displays. The desired result of only displaying one SSID is the same. Furthermore, since the password is the exact same and the victim machine had already been connected to the router once, it used the saved password associated with the SSID in its settings to connect to automatically connect to attacker's machine instead of the router. Figure 8 shows the attacker machine is providing a

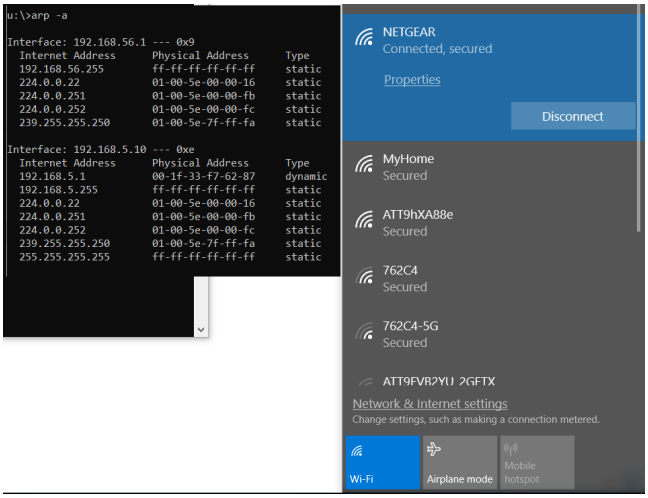

Figure 6: Victim machine under no attack. IP, MAC, and SSID are all from the router.

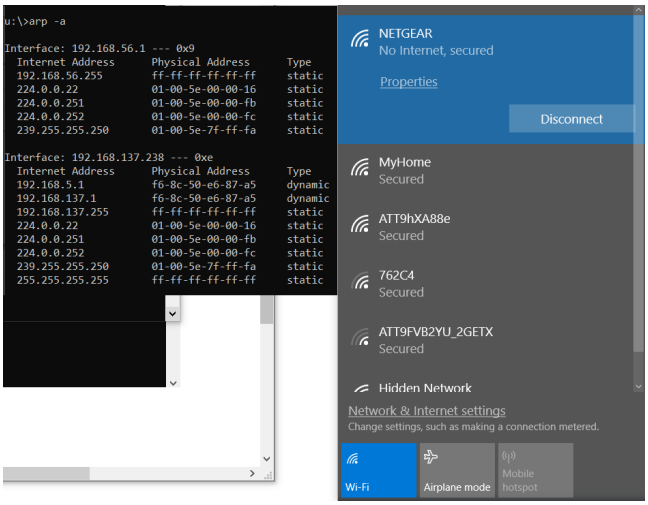

Figure 7: Victim machine is detecting spoofed router, not the original router.

connection to the victim machine. Notice in this figure how the SSID and password are set to the same as the router's SSID and password.

At this point, the remaining tests are deemed failed. Taking a closer look at the victim's ARP table shows that the updated MAC address is not updated. After some research, it was found that ethernet network interface cards on Windows machines are not locked to certain MAC addresses, however, WiFi network interface cards are locked to broadcasting a MAC address that begins with '02' or the default MAC. If a MAC address that does not begin with the octet '02' is set by an administrator, the new MAC address will be ignored and the default will be broadcast to other machines. This is because the '02' signifies that the MAC address was locally changed. Thus, in figure 9 the gateway address is set the attacker's original

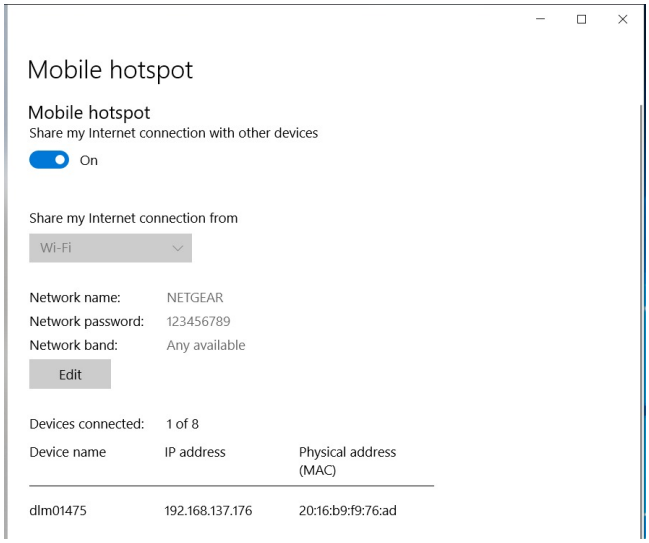

Figure 8: Attacker machine broadcasting WiFi signal, and one victim machine has connected.

WiFi network MAC address. Even after using an ethernet port to directly connect the attacker machine to the router to bypass the '02' octet rule, the hotspot option is broadcast using WiFi network card not the ethernet network card. Thus, this did not help and still the WiFi network card's MAC could not be set and broadcast to other devices by the attacker.

We did notice something odd, however. Normally, if two identical SSIDS are broadcast from two devices that have differing MAC addresses, the available connections panel should display one of the names with a numeric value after it. An example would be "NETGEAR" and "NETGEAR 2". However, this did not happen as shown in the previous figures. After taking a closer look at the victim's ARP table, figure 9, notice that the attacker's MAC address is set to both the router's original gateway address "192.168.5.1" and the attacker' gateway address "192.168.137.1". How is this possible? According to ARP table ruling, two MACs should never show up in an ARP table at the same time.

Our best conjecture would be the fact that since we set the MAC address of the Ethernet port on the attacker's machine to the router's MAC address and then connected to the router, this could have confused the router as well as other devices that were connected to the same network as the attacker and the router (i.e. the victim). Broadcasts for the same MAC for both IPs could have occured, and considering they are gateway addresses, they may have not been filtered correctly. We also verified that the router only assigns values outside of its gateway address as seen in figure 10, yet, the router still provided connection to the attacker machine. We verified the attacker's ethernet port's gateway IP was set to "192.168.5.1" under the IPV4 configuration settings in Windows. Also, the victim machine automatically connected to the attacker's hotspot wifi. The difference in the broadcasted MAC should have prohibited windows from trying the saved "NETGEAR" SSID and "123456789" password combination on the attacker's wifi because these should have been associated with the router's MAC. Again,

```
Interface: 192.168.137.238 --- 0xe
  Internet Address      Physical Address      Type
  192.168.5.1           f6-8c-50-e6-87-a5     dynamic
  192.168.137.1         f6-8c-50-e6-87-a5     dynamic
  192.168.137.255       ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

Figure 9: ARP Cache Poisoned on victim machine with gateway addresses set to attacker's real MAC address

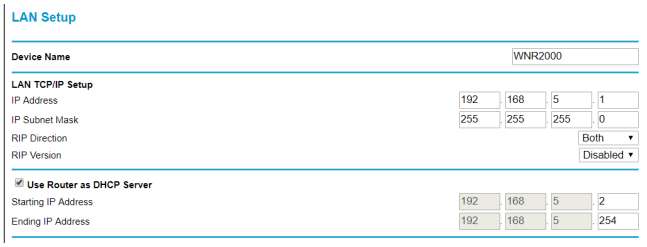

Figure 10: Router configuration set during experimentation.

this could have been a confusion based on the fact that the ARP table MAC was updated to the attacker's MAC over the router's MAC. These unexplained results should be due to a security flaw in routers conductibility to not properly handle a case of an IP and MAC equivalent to its own, and/or Windows security.

After testing several devices, any old or new device would immediately connect to the attacker's SSID "NETGEAR" and not even show the router's SSID. This is a flaw because if we were able to try other methods and perhaps find a way to change the WiFi MAC address to the router's, this would become an almost flawless attack if computer's are not even showing the router's broadcast and the router continuously serves the hot spot an internet connection.

Lastly, this experiment provided one benefit of Windows security features. Windows is a very popular operating system utilized by millions of people across the globe. The built in '02' octet feature may have been installed so people are not able to make configuration changes as attempted here. After some research, it was found that setting the MAC address to begin with a '00' is possible via a USB WiFi dongle.[1] In this experiment, that could have worked considering the dongle could have been utilized by the hotpsot feature to broadcast the WiFi signal over instead of the built in WiFi interface card. Unfortunately, this resource was not available to us at the time of experiment, but will make a good test for the future.

4 DISCOVERY

The goal of this project was to find security vulnerabilities in everyday computing machines, thus a normal household router was utilized. Also, it should be noted that promiscuous mode was not utilized in any of the security approaches. Promiscuous mode would have simply allowed the attacker to view all packets in and out of the router. However, we wanted to act like another user on the network, not simply steal messages. Stealing an identity on a network is far more contagious to a network than simply sniffing packets because many threatening actions can be taken and the blame can be placed on the victim not the attacker.

In the first experiment we found discovered that MAC addresses cannot duplicate in the same network. However, using this characteristic of the router, we can perform a denial of service attack to devices that are currently connected to the network and also successfully use their identity at will. Using the same principle, if we have a reliable way to cut internet connection to the target computer, we can use that period of time to listen to the data transmission from router. By the time the victim came back online and (if) noticed that their identity was stolen, data could already have been taken or actions could already have been completed by the attacker. It would be too late by then. This is especially true when the user isn't even online when their identity is stolen since their identity can be used and then released back to the network for the victim to use at a later time. This process will temporarily remove access from the victim to the network. From a behavioral viewpoint, when a single user cannot connect to the internet, one of the suggestion from technical support is restarting the computer. This gives enough time to remove the right user from the network so the attacker can take the place of the true user.

In the second experiment, it was found that static IP assignment will trump dynamic IP assignment even if that IP is currently in use by another device. The static IP address in this section was set from the attacker machine, not the router configuration. This is an important note because a more secure system could allow static IP set by the device to only go into effect if the device's MAC has a static IP rule set in the router configuration already or if the IP address is not in use. A use of this security flaw would of course be to accept messages intended for another user at that IP address. If there is an unmonitored IoT device on a network, an attacker could possibly steal the IP address of that device. When a remote user trys to access the device by referencing the IP address, they will be connected to the attackers device not the IoT device.

The third experiment showed promsing results in the beginning, but later security protocols within Windows operating system kicked in. Although a dissapointment for us that we could not succeed with our attack, this was a great move that the operating system made to ensure that an attacker could not hijack the entire network. However, we did successfully confuse either the operating system, router, or possibly both when stealing the router's IP and MAC addresses. Given enough time, this attack may actually be possible, espeically given a WiFi USB dongle as suggested in the security concerns section of the third experiment.

4.1 Limitations

A few limitations to mention are that the setting up and testing of experiments done in this approach were based on a messaging script that blindly only required an IP address to send plain text messages. In real world scenario, these messages would likely be encrypted and other mechanisms may have to be overcome for a real world attack to complete. However, the purpose of the experiments were meant for an malicious "insider" to create these attacks. None of the

attacks aforementioned would have applied if the attacker wasn't even able to connect to the router in the first place. Many companies also do not provide admin access to their employees on network devices, perhaps so attacks such as the ones in this report cannot be done. The user must be in admin mode to change the MAC address, set a static IP address, or even turn on the hotspot feature within Windows.

Additionally, the router utilized in the attacks was almost 10 years old. It is possible that some newer models could provide better security, however, there are thousands of people that still use older models of routers as they essentially all do the same function at the end of the day still: provide internet connection. For this reason, manufacturers should consider sending out updates to older models even if they have fixed these issues in newer models.

5 CONCLUSION

It was far too simple to complete all three of the experiments mentioned in this report. We had initially thought that there would be security protocols within routing systems to prevent some of the attacks, but really once an attacker is into a network they can perform an outrageous number of attacks. With modern day technology, this should not be happening and manufacturers should be making updates as more attacks are found. We successfully found new methods for denial of service and identity theft in this project, and we hope to research and report back more in the future.

Additionally, with more time, we want to look into the Aircrack-ng tool to send disassociate packages to a client currently connected to an access point. According to their website, it is possible to force change an access point for a client, so we can overcome the difficulty of MAC conflict when spoofing MAC address. By removing the client from the network, it is possible to pretend to be a client with the same MAC address without alerting the client of the intermittent connection. Instead of using drones in the Skyjack project, we can use a laptop in a busy location to test out this tool. [4]

REFERENCES

[1] [n. d.]. How to bypass first octet restriction for wireless adapter mac addresses in windows 8.1. ([n. d.]). Retrieved May 09, 2019 from https://superuser.com/questions/1265544/how-to-bypass-first-octet-restriction-for-wireless-adapter-mac-addresses-in-wind

[2] [n. d.]. ifconfig(8) - Linux man page. ([n. d.]). Retrieved April 26, 2019 from https://linux.die.net/man/8/ifconfig

[3] ADMINISTRATOR. [n. d.]. MEDIA ACCESS CONTROL - MAC ADDRESSES. ([n. d.]). Retrieved April 25, 2019 from http://www.firewall.cx/networking-topics/general-networking/105-mac-addresses.html

[4] Aircrack-ng. [n. d.]. Aircrack - Deauthentication. ([n. d.]). Retrieved May 09, 2019 from http://www.aircrack-ng.org/doku.php?id=deauthentication

[5] Edgar D. Cardenas. [n. d.]. MAC Spoofing – An Introduction. ([n. d.]). Retrieved April 25, 2019 from https://www.giac.org/paper/gsec/3199/mac-spoofing-an-introduction/105315

[6] CHRIS HOFFMAN. [n. d.]. How (and Why) to Change Your MAC Address on Windows, Linux, and Mac. ([n. d.]). Retrieved April 25, 2019 from https://www.howtogeek.com/192173/how-and-why-to-change-your-mac-address-on-windows-linux-and-mac/

[7] Zeus Kerravala. [n. d.]. DHCP defined and how it works. ([n. d.]). Retrieved May 09, 2019 from https://www.networkworld.com/article/3299438/dhcp-defined-and-how-it-works.html

[8] Alvaro Lopez. [n. d.]. MAC Changer. ([n. d.]). Retrieved April 25, 2019 from http://manpages.org/macchanger