Flying Access Points

January 10, 2016

1 Captive Portal

1.1 Introduction to Captive Portals

It is very common for WiFi networks to have some sort of identification of users. A normal way of doing this is by creating a captive portal. A captive portal is when users connecting to the network are redirected to a login screen, which normally is a website hosted by the WiFi provider. This allows the provider to restrict the use of the network to authorized users and to gather information about which users use the network and when.

1.2 The work process

We decided to implement a captive portal through the use of iptables. Iptables is a flexible firewall utility that uses policy chains to allow or block traffic [1]. It allows you to create a list of rules which all incoming traffic will be checked against. Should a rule be applicable it will be applied. If no rule is found the default action will be taken[1].

We began by defining the requirements needed to create the captive portal. From that we arrived at an initial set of requirements which were later expanded upon as our knowledge of iptables increased which allowed us to better understand what was needed to achieve the desired results. Based on these requirements we designed the following rules to be implemented:

1. iptables -N internet -t mangle;
2. iptables -t mangle -A PREROUTING -j internet;
3. iptables -t mangle -A internet -m mac –mac-source a2:ee:d5:f3:9e:14 -j RETURN;
4. iptables -t mangle -A internet -j MARK –set-mark 99;
5. iptables -t nat -A PREROUTING -m mark –mark 99 -p tcp –dport 80 -j DNAT –to 192.168.43.1:8080;
6. iptables -t filter -A FORWARD -m mark –mark 99 -j DROP;
7. iptables -t filter -A INPUT -d 192.168.43.1 -p tcp –dport 80 -j ACCEPT;
8. iptables -t filter -A INPUT -p tcp –dport 80 -j ACCEPT;

9. iptables -t filter -A INPUT -p udp –dport 53 -j ACCEPT;
10. iptables -t filter -A INPUT -m mark –mark 99 -j DROP;
11. iptables -A FORWARD -i p2p0 -o wlan0 -m state –state ESTABLISHED,RELATED -j ACCEPT;
12. iptables -A FORWARD -i wlan0 -o p2p0 -j ACCEPT;
13. iptables -t nat -A POSTROUTING -o p2p0 -j MASQUERADE;

We then went on to create a small app which runs these rules on the Sony Xperia we were working on. This was the step that took the most time as we had several issues during the process.

Finally we attempted to create the captive portal itself through the use of a local website on the Xperia through the use of nanohttpd[2]. This is where we ran out of time and as such we did not have the ability to create a proper login screen. Instead we decided to focus our efforts on proving that the redirect was functioning properly and as such made a website which only displays a small amount of text to confirm that we have been properly redirected.

1.3 Issues

During our work we ran into several obstacles that halted our progress. The greatest of these was testing our code. This was a very time-consuming task which led to slow progress. The reason why this took unexpectedly much time was because it was hard to debug errors due to many Linux libraries being used which caused errors with unclear reasons and also the rules themselves could not be tested until the program itself was functional. The error we ran into the most was an exception caused by a "broken pipe" which was a lost connection. This happened while running the initialization process where all rules were created. Solving this took several days. We eventually found a way to avoid this issue by having the rules stored in an external file saved to the SD-card of the phone instead of having them defined in the code.

Another issue which caused great delays was building an understanding of iptables and how to use them. This took far more time than expected. In the end the cumulative delays prevented us from creating the logic logic for the captive portal.

1.4 Results

Due to large delays in our progress we were unable to completely finish the captive portal but we have managed to create most of it. We have the ability to redirect incoming traffic to a website located on the Sony Xperia. This website however only contain a few lines of text to confirm that the redirect worked as intended.

We have implemented some of the methods needed to implement the login logic but have been unable to properly test them. We have also come to change

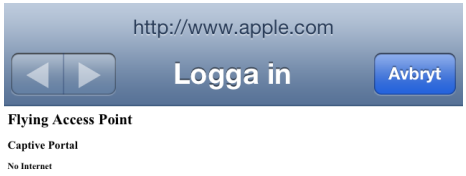

Figure 1: User Screenshot

one of our design choices over the course of our work. Due to this we are not using all of the rules. We have removed the following rules:

1. iptables -t mangle -A internet -m mac –mac-source a2:ee:d5:f3:9e:14 -j RETURN;

2. iptables -t filter -A INPUT -m mark –mark 99 -j DROP;

The first rule was removed due to us not having been able to finish the captive portal. It is the rule which, based on a list of mac addresses, decides whether or not a connecting user shall be allowed to access the internet directly. The second rule was intended to block traffic that was not directed at any of the ports we were using but we removed it as we found it to be better not to block all other traffic.

Now we will explain the remaining rules:

1. iptables -N internet -t mangle;
   Creates a new chain named ”internet” in the manging table (iptable). The purpose of this table is to manipulate packets before the routing decision is made.

2. iptables -t mangle -A PREROUTING -j internet;
   The ”-A” modifier signifies that a pre-routing rule is appended to the table mangle and to our chain ”internet”. The ”-t” specifies the table, which is ”mangle”

3. iptables -t mangle -A internet -j MARK –set-mark 99;
   Append a rule to our chain to mark all incoming packets with a 99 mark

4. iptables -t nat -A PREROUTING -m mark –mark 99 -p tcp –dport 80 -j DNAT –to 192.168.43.1:8080;
   Append a rule to the PREROUTING chain in the NATing table, that any packet that has been marked with 99 and has the destination tcp port 80, will go through a static nating redirecting it to given ip and port, that ip and port in our case here is the phone ip where our nano webserver is running on port 8080

5. iptables -t filter -A FORWARD -m mark –mark 99 -j DROP;
   Append to the forward chain (which is packets being forwarded) a rule to drop all packets marked with mark 99. This means that all packets we just redirected in the past rule will be stopped by this rule so that they don’t do any further.

6. iptables -t filter -A INPUT -d 192.168.43.1 -p tcp –dport 80 -j ACCEPT;
   Append to the input chain (all packets coming in to the phone ip) a rule to accept all packets coming to port 80 (accessing www) which have not been marked by the 99 mark

7. iptables -t filter -A INPUT -p tcp –dport 80 -j ACCEPT;
   Append to the input chain a rule to accept all packets coming to port 80 (accessing wwww) which are not marked.

8. iptables -t filter -A INPUT -p udp –dport 53 -j ACCEPT;
   Does the same as rule 7, only for port 53 (which is DNS).

9. iptables -A FORWARD -i p2p0 -o wlan0 -m state –state ESTABLISHED,RELATED -j ACCEPT;
   Accept forwarding from input interface p2p0 (the android 3g/4g interface) to the output interface WiFi packets with tcp state established. That state means the connection that has been already established (meaning the packets succeeded to reach the internet)

10. iptables -A FORWARD -i wlan0 -o p2p0 -j ACCEPT;
    Accepts forwarding in the other direction, complimenting rule 9.

11. iptables -t nat -A POSTROUTING -o p2p0 -j MASQUERADE;
    This rule does the dynamic nating on all packets going out to the internet (p2p0) in the postrouting chain.

The "-A" marker found in the rules above stands for "Append" and is used to append new rules to the chain. When disabling the captive portal the same rules as above are executed with the exception of exchanging all "-A" for "-D" which instructs the app to delete the rules instead.

- Overview of the buttons seen in the app screenshot in Figure 2

Rules This button in the app displays the rules currently in use on the screen.

Captive Portal Enables/Disables the captive portal.

Webserver Starts/Stops the nanohttpd webserver which the captive portal redirects to. If it is disabled the users will be able to access the internet directly.

Hotspot Enables/Disables the WiFi hotspot.

References

[1] Brown, K. (2014, June 2). The Beginner's Guide to iptables, the Linux Firewall. `http://www.howtogeek.com/177621/the-beginners-guide-to-iptables-the-linux-firewall/`. Accessed: 2015-12-20.

[2] Tiny, easily embeddable HTTP server in Java, `https://github.com/NanoHttpd/nanohttpd`. Accessed: 2015-12-20.

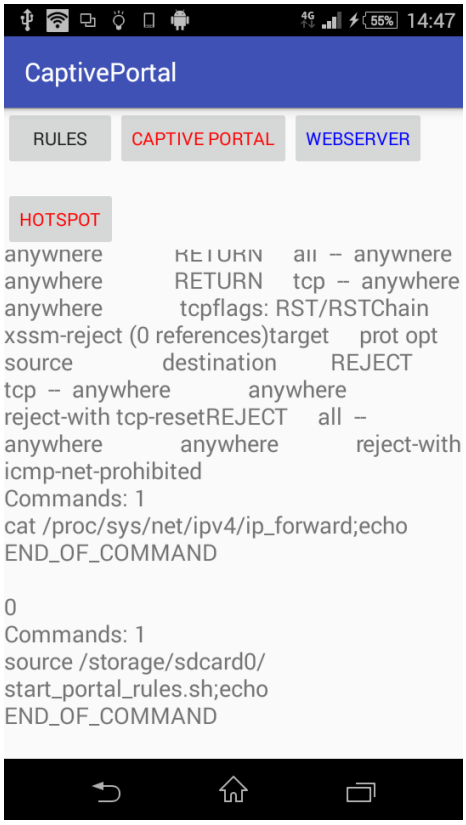

Figure 2: Captive Portal App Screen-shot