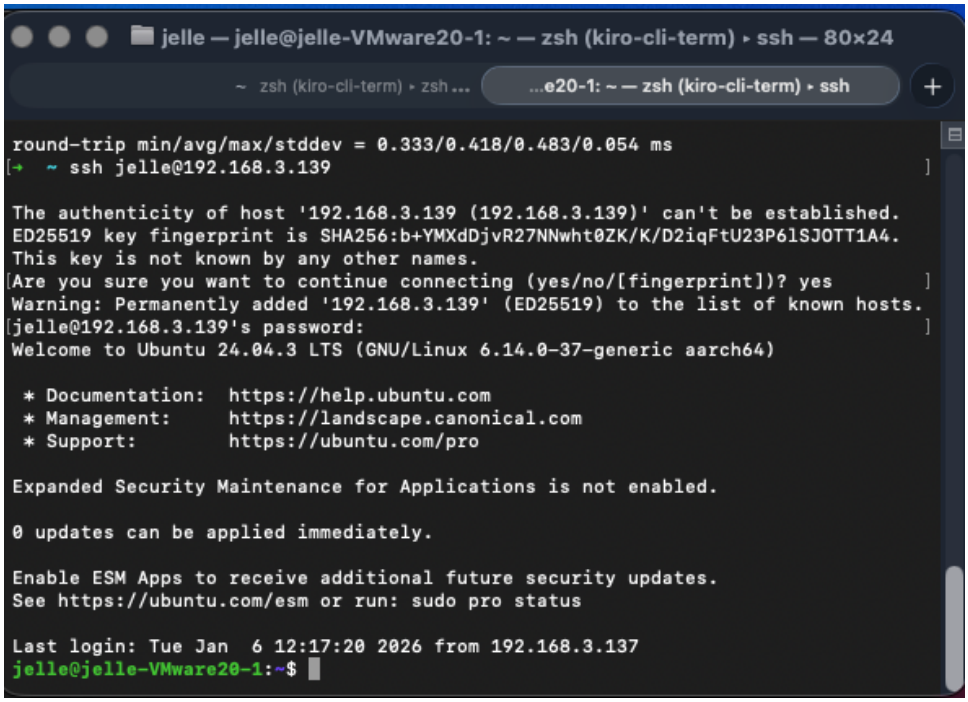
Template Week 6 – Networking

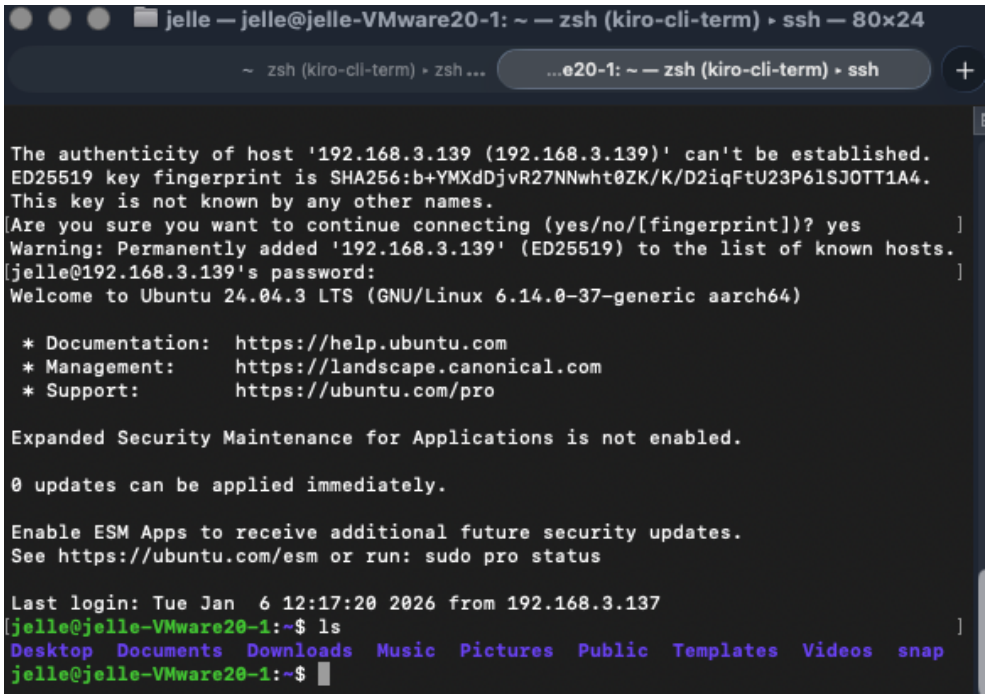
Student number: 589530

Assignment 6.1: Working from home

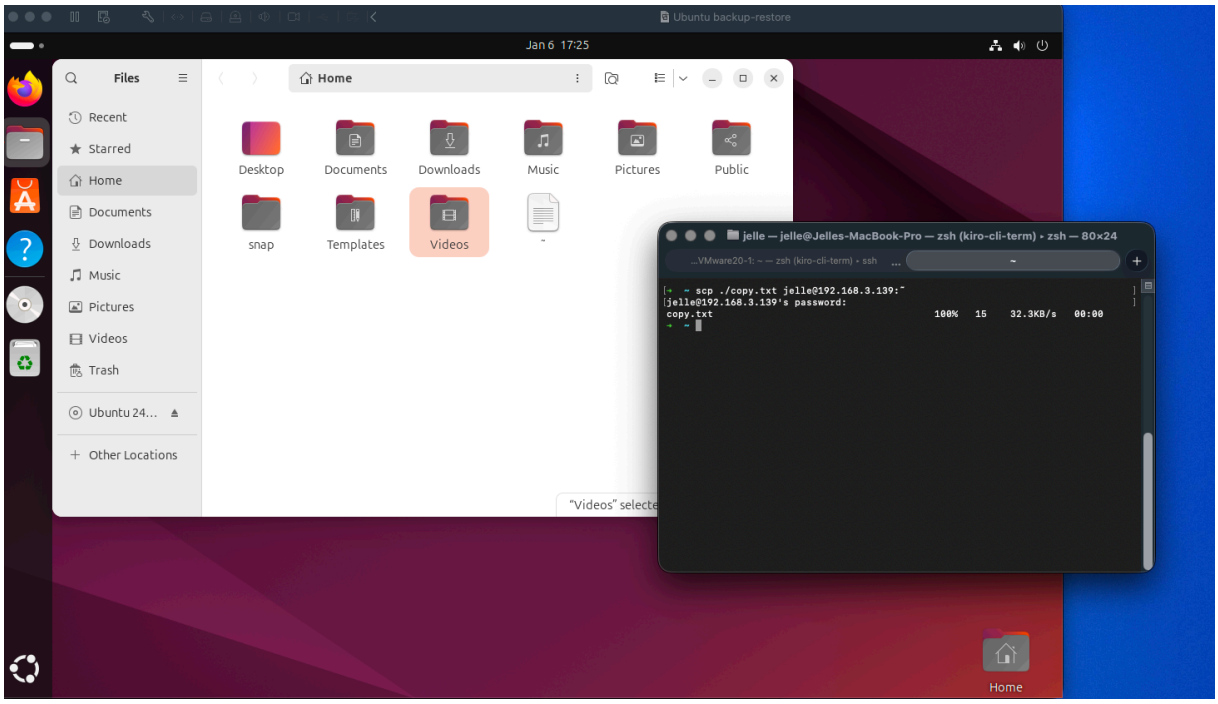
Screenshot installation openssh-server:



Screenshot successful SSH command execution:

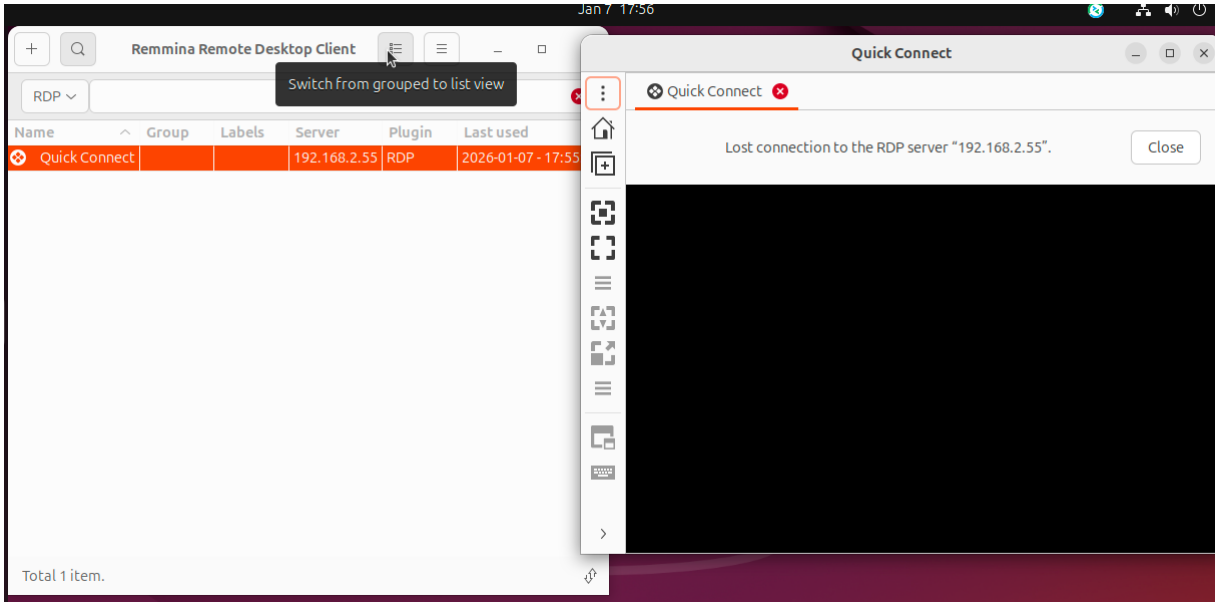


Screenshot successful execution SCP command:



Screenshot remmina:

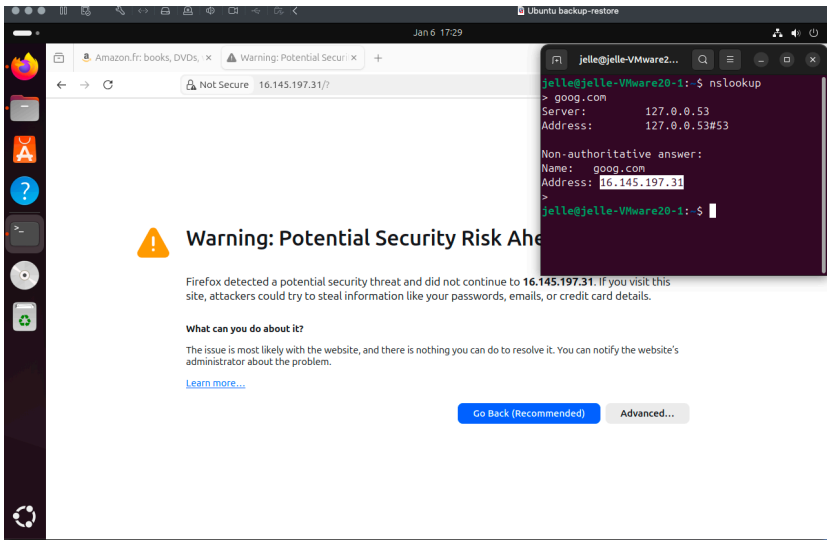
Na lang proberen kreeg ik het niet werkend helaas, aangezien dit anders werkt bij mac.



Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:

Screenshot website visit via IP address:



Assignment 6.3: subnetting

How many IP addresses are in this network configuration 192.168.110.128/25?

/25 betekend: 32 – 25 = 7 en 2 tot de macht 7 = 128, dus er kunnen 128 IP-adressen erop

What is the usable IP range to hand out to the connected computers?

192.168.110,128 tot 192.168.110.255 maar je moet ook altjid 1 netwerk en braodcat ipaddress heben dus eigenlijk zijn het er 126 met een range van: 192.168.110.129 – 192.168.110.254

Check your two previous answers with this Linux command: `ipcalc 192.168.110.128/25`

```
[jelle@jelle-VMware20-1:~$ ipcalc 192.168.110.128/25                          ]
Address:   192.168.110.128      11000000.10101000.01101110.1 0000000
Netmask:   255.255.255.128 = 25 11111111.11111111.11111111.1 0000000
Wildcard:  0.0.0.127            00000000.00000000.00000000.0 1111111
=>
Network:   192.168.110.128/25   11000000.10101000.01101110.1 0000000
HostMin:   192.168.110.129      11000000.10101000.01101110.1 0000001
HostMax:   192.168.110.254      11000000.10101000.01101110.1 1111110
Broadcast: 192.168.110.255      11000000.10101000.01101110.1 1111111
Hosts/Net: 126                   Class C, Private Internet
```

Explain the above calculation in your own words.

Address: hexadecimal naar binary

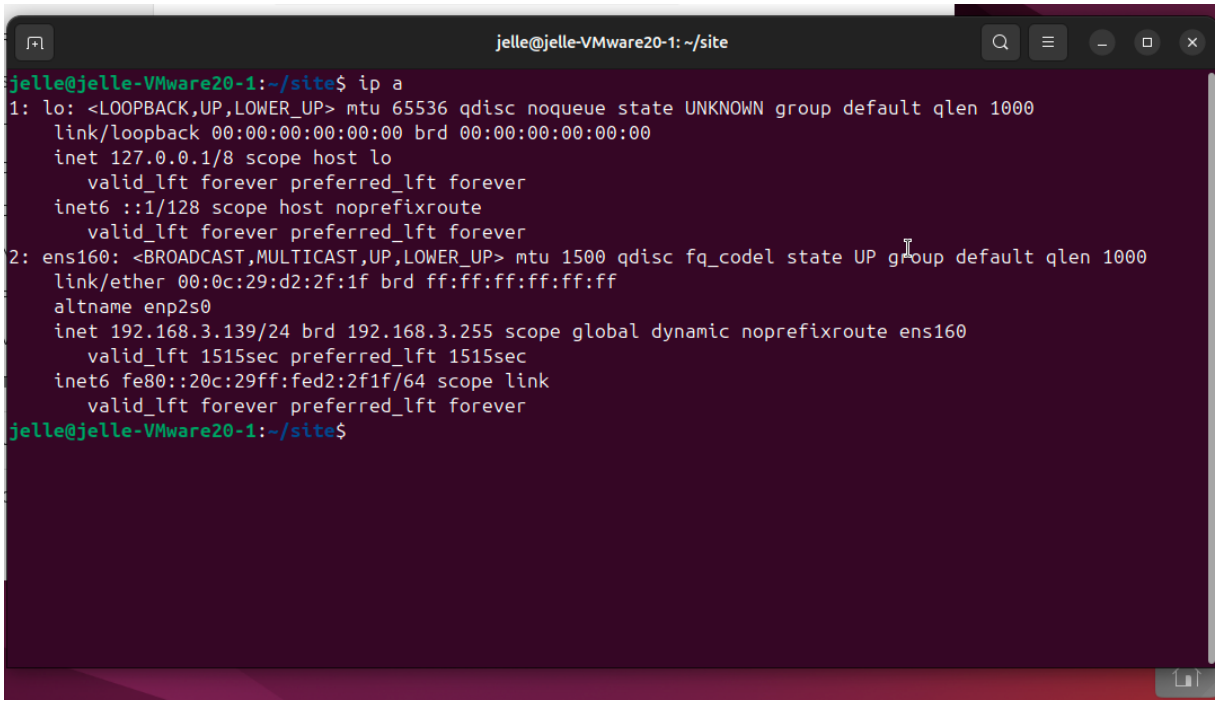
Netmask: hexadecimal naar binary, en de laatste stuk is 0, die zijn bruikbaar, verder omdat die allemaal bruikbaar zijn kan je ook de Host/ Network verklaren aangezien hij in klasse C zit. Aangezien je alleen de laatste stuk mag gebruiken.

Verder hoe er bij de network bij bijgekomen is door een OR te gebruiken op de Address en de Netmask.

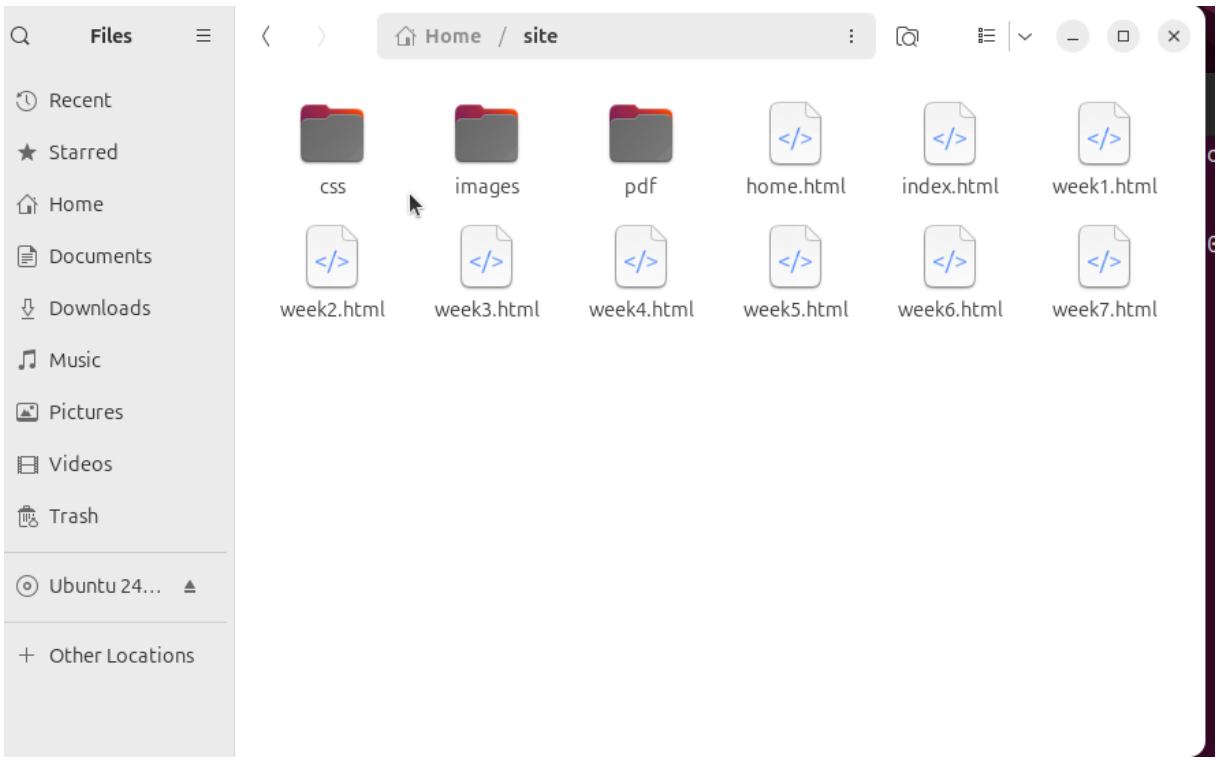
Host/Net: aantal bruikbare ip adressen.

Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:



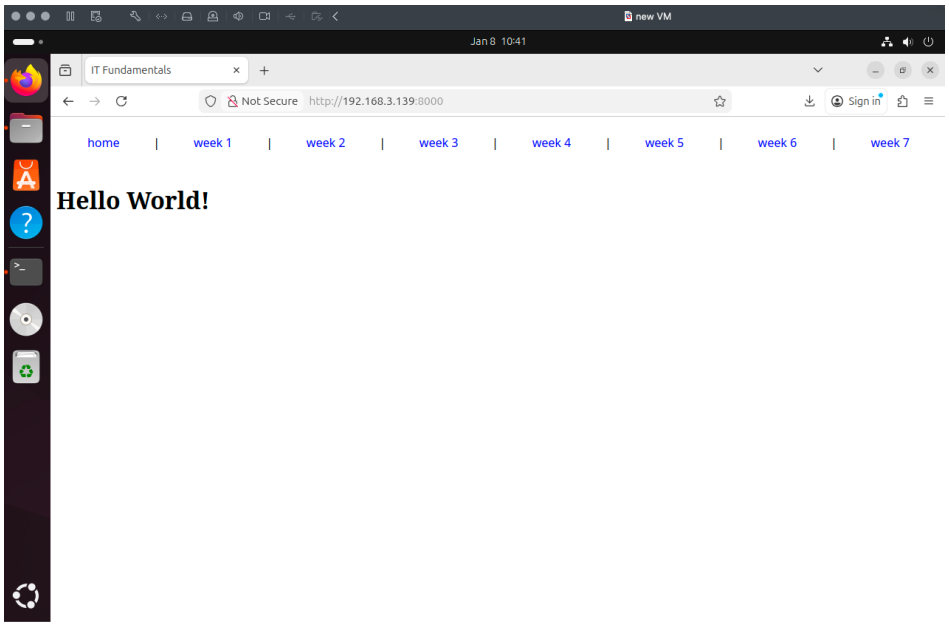
Screenshot of Site directory contents:



Screenshot python3 webserver command:

```
jelle@jelle-VMware20-1:~/site$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
127.0.0.1 - - [08/Jan/2026 10:40:50] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Jan/2026 10:40:50] "GET /css/mypdfstyle.css HTTP/1.1" 200 -
127.0.0.1 - - [08/Jan/2026 10:40:50] "GET /home.html HTTP/1.1" 200 -
127.0.0.1 - - [08/Jan/2026 10:40:50] code 404, message File not found
127.0.0.1 - - [08/Jan/2026 10:40:50] "GET /favicon.ico HTTP/1.1" 404 -
192.168.3.139 - - [08/Jan/2026 10:40:58] "GET / HTTP/1.1" 200 -
192.168.3.139 - - [08/Jan/2026 10:40:58] "GET /home.html HTTP/1.1" 200 -
192.168.3.139 - - [08/Jan/2026 10:40:58] "GET /css/mypdfstyle.css HTTP/1.1" 200 -
192.168.3.139 - - [08/Jan/2026 10:40:58] code 404, message File not found
192.168.3.139 - - [08/Jan/2026 10:40:58] "GET /favicon.ico HTTP/1.1" 404 -
```

Screenshot web browser visits your site



Assignment 6.5: Network segment

Remember that bitwise java application you've made in week 2? Expand that application so that you can also calculate a network segment as explained in the PowerPoint slides of week 6. Use the bitwise & AND operator. You need to be able to input two Strings. An IP address and a subnet.

IP: 192.168.1.100 and subnet: 255.255.255.224 for /27

```
Example: 192.168.1.100/27
Calculate the network segment
IP Address:   11000000.10101000.00000001.01100100
Subnet Mask:  11111111.11111111.11111111.11100000
-----------------------------------------------
Network Addr: 11000000.10101000.00000001.01100000

This gives 192.168.1.96 in decimal as the network address.
For a /27 subnet, each segment (or subnet) has 32 IP addresses (2^5).
The range of this network segment is from 192.168.1.96 to 192.168.1.127.
```

Paste source code here, with a screenshot of a working application.

Ready? Save this file and export it as a pdf file with the name: **week6.pdf**

hello-world.js

```java
import java.util.Scanner;
import java.util.StringTokenizer;

public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        System.out.println("Please enter an number");
        int number = scn.nextInt();

        System.out.println("is the number odd?: " +
isNumberOdd(number));
        System.out.println("is the power of 2?: " +
isNumberPowerOfTwo(number));

        calcNetworkAddress();
    }

    public static boolean isNumberOdd(int number) {
        return (number & 1) == 1;
    }

    public static boolean isNumberPowerOfTwo(int number) {
        return (number & (number - 1)) == 0;
    }

    public static void calcNetworkAddress() {
        String ipString = "192.168.1.100";
        //255.255.255.224
        String subnetString = "255.255.255.224";

        // rewrite the sting, other no longer useful cleaner this
way.
        ipString = stringToBinair(ipString);
        subnetString = stringToBinair(subnetString);
        System.out.println("Ip binary: " + ipString);
        System.out.println("Subnet binary: " + subnetString);
        System.out.println("-----------------------------------
------------");

        String cmpResult = compareBinaryBitwise(ipString,
subnetString);
        System.out.println("netwerk address: " + cmpResult);

    }

    public static String stringToBinair(String str){
        StringTokenizer tokenizer = new StringTokenizer(str, ".");
        StringBuilder binairySting = new StringBuilder();
        while (tokenizer.hasMoreTokens()){
            String octetStr = tokenizer.nextToken();
            int octetInt = Integer.parseInt(octetStr);

            // build string with .
            binairySting.append(Integer.toBinaryString(octetInt));
            binairySting.append(".");
        }
        return binairySting.toString();
    }

    public static String compareBinaryBitwise(String str1, String
str2){
        int amountOfTimes = 3; // 4
        int count;

        // slpit the sting based on .
        String[] str1Split = str1.split("\\.");
        String[] str2Split = str2.split("\\.");

        StringBuilder result = new StringBuilder();
        // for each octet do this:
        for (count = 0; count <= amountOfTimes; count++){
            int a = Integer.parseInt(str1Split[count]);
            int b = Integer.parseInt(str2Split[count]);
            int andResult = a & b;

            // convert to 8 bit binary with 0's
            String binary = String.format("%8s",
Integer.toBinaryString(andResult)).replace(' ', '0');
            result.append(binary);
            count++;
        }
        return result.toString();
    }
}

```