

Template Week 6 – Networking

Student number: 590620

Assignment 6.1: Working from home

Screenshot installation openssh-server:

```
josh@josh-VMware-Virtual-Platform:~$ sudo apt install openssh-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-client openssh-sftp-server ssh-import-id
Suggested packages:
  libbsd-dev monkeysphere ssh-askpass molly-guard
  openssh-client
The following packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
The following packages will be upgraded:
  openssh-client
1 upgraded, 4 newly installed, 0 to remove and 156 not upgraded.
Need to get 1,738 kB of archives.
After this operation, 6,743 kB of additional disk space will be used.
Get:1 http://nl.archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-client
amd64 1:9.6p1-3ubuntu13.14 [906 kB]
Get:2 http://nl.archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-sftp-
```

Screenshot successful SSH command execution:

```
josh@josh-VMware-Virtual-Plati X + v
Host key verification failed.
C:\Users\Josh>ssh josh@192.168.88.128
The authenticity of host '192.168.88.128 (192.168.88.128)' can't be established.
ED25519 key fingerprint is SHA256:7IjxsJ2RLi7IBNr+7o803pLkEnPW0izzVeC7Tfdbnwk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.88.128' (ED25519) to the list of known hosts.
josh@192.168.88.128's password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-37-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

157 updates can be applied immediately.
6 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

6 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
```

Screenshot successful execution SCP command:

```
C:\Users\Jos>scp testfile.txt jos@192.168.88.128:/home/jos/
jos@192.168.88.128's password:
testfile.txt
100% 22 10.7KB/s 00:00

C:\Users\Jos>ssh jos@192.168.88.128
jos@192.168.88.128's password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-37-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

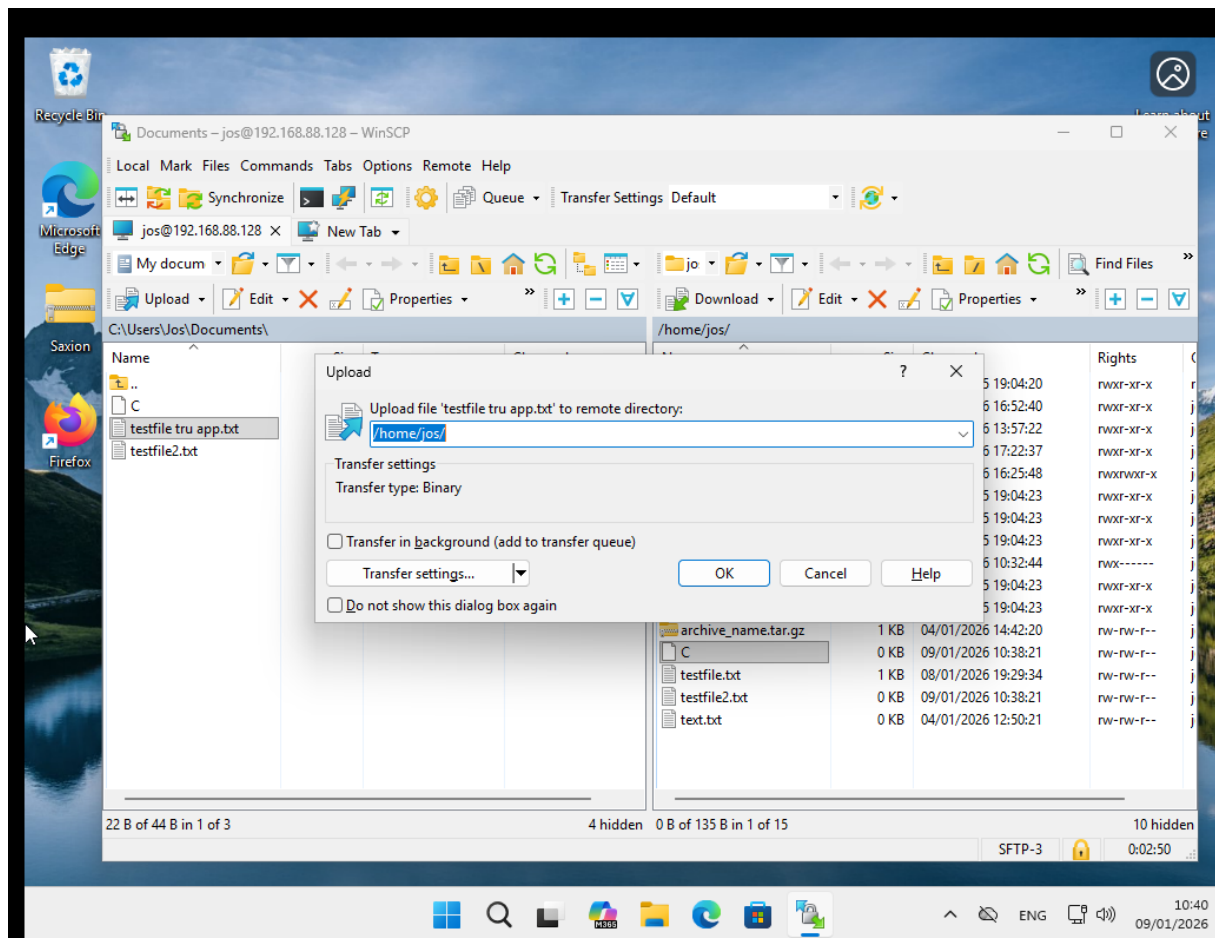
Expanded Security Maintenance for Applications is not enabled.

157 updates can be applied immediately.
6 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

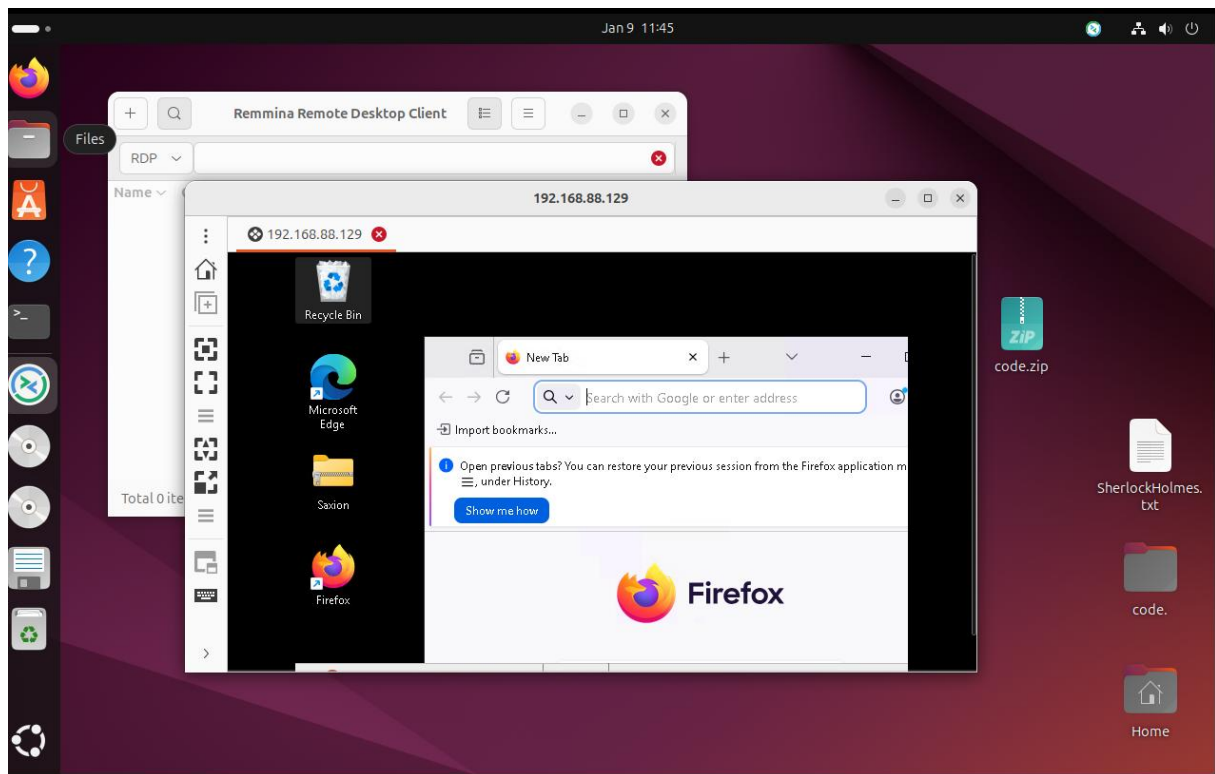
6 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Last login: Thu Jan  8 20:25:42 2026 from 192.168.88.1
jos@jos-Virtual-Platform:~$ ls
archive_name.tar.gz Desktop Documents Downloads hello Music Pictures Public snap Templates testfile.txt text.txt Videos
jos@jos-Virtual-Platform:~$
```

Of via de app



Screenshot remmina:



Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:

Microsoft Windows [Version 10.0.26100.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Jos>nslookup

Default Server: one.one.one.one
Address: 1.1.1.1

> amazon.com

Server: one.one.one.one
Address: 1.1.1.1

Non-authoritative answer:

Name: amazon.com
Addresses: 98.87.170.74
98.87.170.71
98.82.161.185

> google.com

Server: one.one.one.one
Address: 1.1.1.1

Non-authoritative answer:

Name: google.com
Addresses: 2a00:1450:400e:803::200e
216.58.208.110

> one.one.one.one

Server: one.one.one.one
Address: 1.1.1.1

Non-authoritative answer:

Name: one.one.one.one
Addresses: 2606:4700:4700::1001
2606:4700:4700::1111
1.1.1.1
1.0.0.1

> dns.google.com

Server: one.one.one.one
Address: 1.1.1.1

Non-authoritative answer:

Name: dns.google.com
Addresses: 2001:4860:4860::8844
2001:4860:4860::8888
8.8.8.8
8.8.4.4

> bol.com

Server: one.one.one.one
Address: 1.1.1.1

Non-authoritative answer:

Name: bol.com
Address: 79.170.100.62

> w3schools.com

Server: one.one.one.one
Address: 1.1.1.1

Non-authoritative answer:

Name: w3schools.com
Addresses: 13.248.240.135
76.223.115.82

> exit

C:\Users\Jos>

Screenshot website visit via IP address:



Assignment 6.3: subnetting

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

$$32 - 25 = 7$$

$$2^7 = 128$$

$$128 - 2 = 126 \text{ bruikbare IP adressen}$$

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

$$192.168.110.129 - 192.168.110.254$$

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

```
joe@joe-VMware-Virtual-Platform:~$ 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.

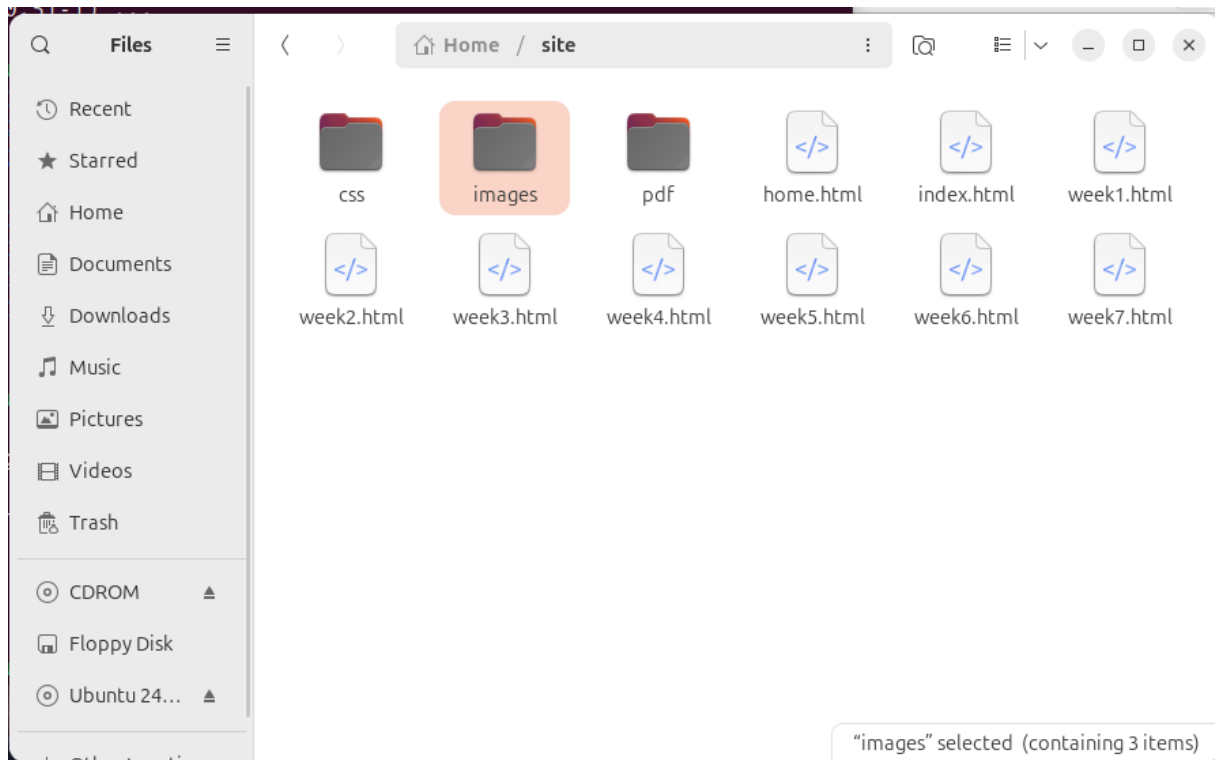
Een /25-subnet splitst een Class C-netwerk op in twee gelijke delen.

Elk deel bevat 128 want $256/2$. Het eerste adres (192.168.110.128) is het netwerkadres en wordt gebruikt om het netwerk zelf aan te duiden. Dit adres kan niet aan apparaten worden toegewezen. Het laatste adres (192.168.110.255) is het broadcastadres en is gereserveerd. Hierdoor blijven 126 bruikbare IP-adressen over die kunnen worden toegewezen aan apparaten. Deze lopen van 192.168.110.129 tot en met 192.168.110.254.

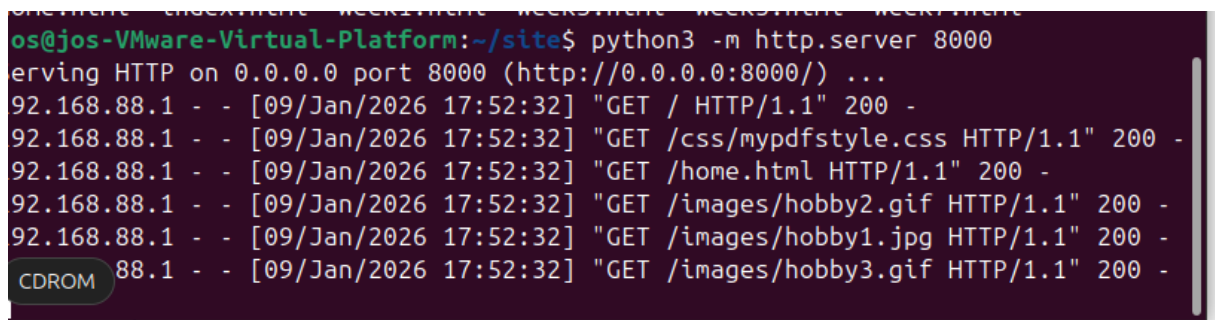
Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

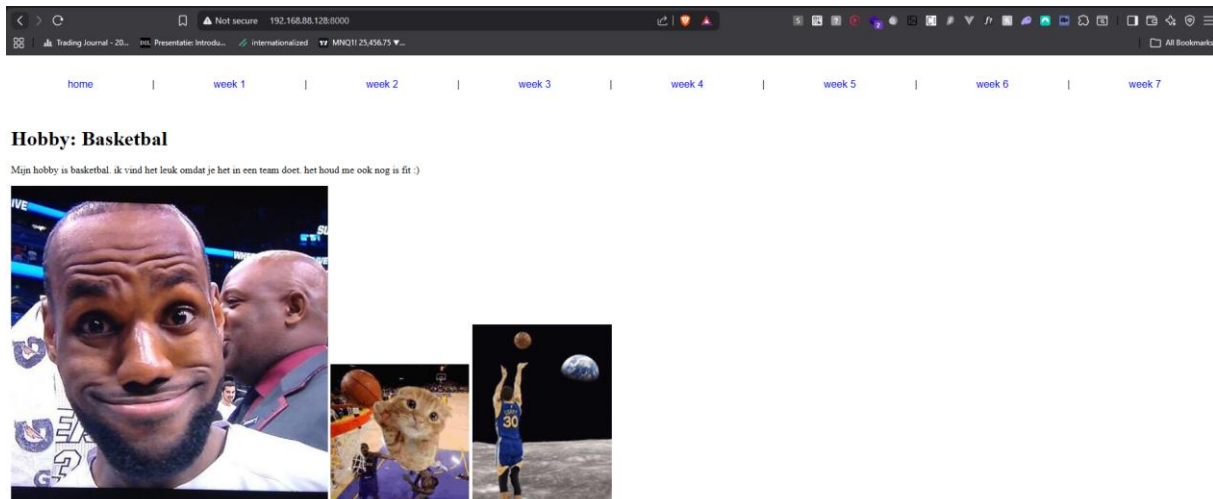
Screenshot of Site directory contents:



Screenshot python3 webserver command:



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.

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

```
public class Main {
```

```
    public static void main(String[] args) {
```

```

Scanner scanner = new Scanner(System.in);

System.out.println("\n--- MENU ---");
System.out.println("1. Is number odd?");
System.out.println("2. Is number a power of 2?");
System.out.println("3. Twos complement of number");
System.out.println("4. Calculate network segment (IP & Subnet)");
System.out.print("Choose an option (1-4): ");

int choice = scanner.nextInt();
scanner.nextLine(); // consume newline

switch (choice) {
    case 1:
        System.out.print("Enter a number: ");
        int number1 = scanner.nextInt();
        checkOdd(number1);
        break;

    case 2:
        System.out.print("Enter a number: ");
        int number2 = scanner.nextInt();
        checkPowerOfTwo(number2);
        break;

    case 3:
        System.out.print("Enter a number: ");
        int number3 = scanner.nextInt();
        twosComplement(number3);
        break;

```


case 4:

```
System.out.print("Enter IP address (For exemple 192.168.1.100): ");
```

```
String ip = scanner.nextLine();
```

```
System.out.print("Enter subnet mask (For exemple 255.255.255.224): ");
```

```
String subnet = scanner.nextLine();
```

```
calculateNetwork(ip, subnet);
```

```
break;
```

default:

```
System.out.println("Invalid choice.");
```

```
}
```

```
scanner.close();
```

```
}
```

```
public static void checkOdd(int number) {
```

```
    if ((number & 1) == 1)
```

```
        System.out.println(number + " is odd");
```

```
    else
```

```
        System.out.println(number + " is even");
```

```
}
```

```
public static void checkPowerOfTwo(int number) {
```

```
    if (number > 0 && (number & (number - 1)) == 0)
```

```
        System.out.println(number + " is a power of 2");
```

```
    else
```

```
        System.out.println(number + " is NOT a power of 2");
```

```
}
```

```

public static void twosComplement(int number) {
    int twosComp = ~number + 1;
    System.out.println("Twos complement of " + number + " = " + twosComp);
}

```

```

public static void calculateNetwork(String ip, String subnet) {

```

```

    int[] ipParts = parseAddress(ip);
    int[] subnetParts = parseAddress(subnet);
    int[] networkParts = new int[4];

```

```

    System.out.println("\nIP Address (binary):");
    printBinary(ipParts);

```

```

    System.out.println("Subnet Mask (binary):");
    printBinary(subnetParts);

```

```

    System.out.println("-----");

```

```

    for (int i = 0; i < 4; i++) {
        networkParts[i] = ipParts[i] & subnetParts[i];
    }

```

```

    System.out.println("Network Address (binary):");
    printBinary(networkParts);

```

```

    System.out.println("\nNetwork Address (decimal): "
        + networkParts[0] + "." + networkParts[1] + "."
        + networkParts[2] + "." + networkParts[3]);

```

```

    int blockSize = 256 - subnetParts[3];

```

```

int start = networkParts[3];

int end = start + blockSize - 1;


System.out.println("Network range:");

System.out.println(ipParts[0] + "." + ipParts[1] + "." + ipParts[2] + "." + start +
    " - " +
    ipParts[0] + "." + ipParts[1] + "." + ipParts[2] + "." + end);
}


private static int[] parseAddress(String address) {
    String[] parts = address.split("\\.");
    int[] nums = new int[4];
    for (int i = 0; i < 4; i++) {
        nums[i] = Integer.parseInt(parts[i]);
    }
    return nums;
}


private static void printBinary(int[] parts) {
    for (int i = 0; i < parts.length; i++) {
        System.out.print(String.format("%8s",
            Integer.toBinaryString(parts[i])).replace(' ', '0'));
        if (i < parts.length - 1)
            System.out.print(".");
    }
    System.out.println();
}
}

```

Ready? Save this file and export it as a pdf file with the name: [week6.pdf](#)