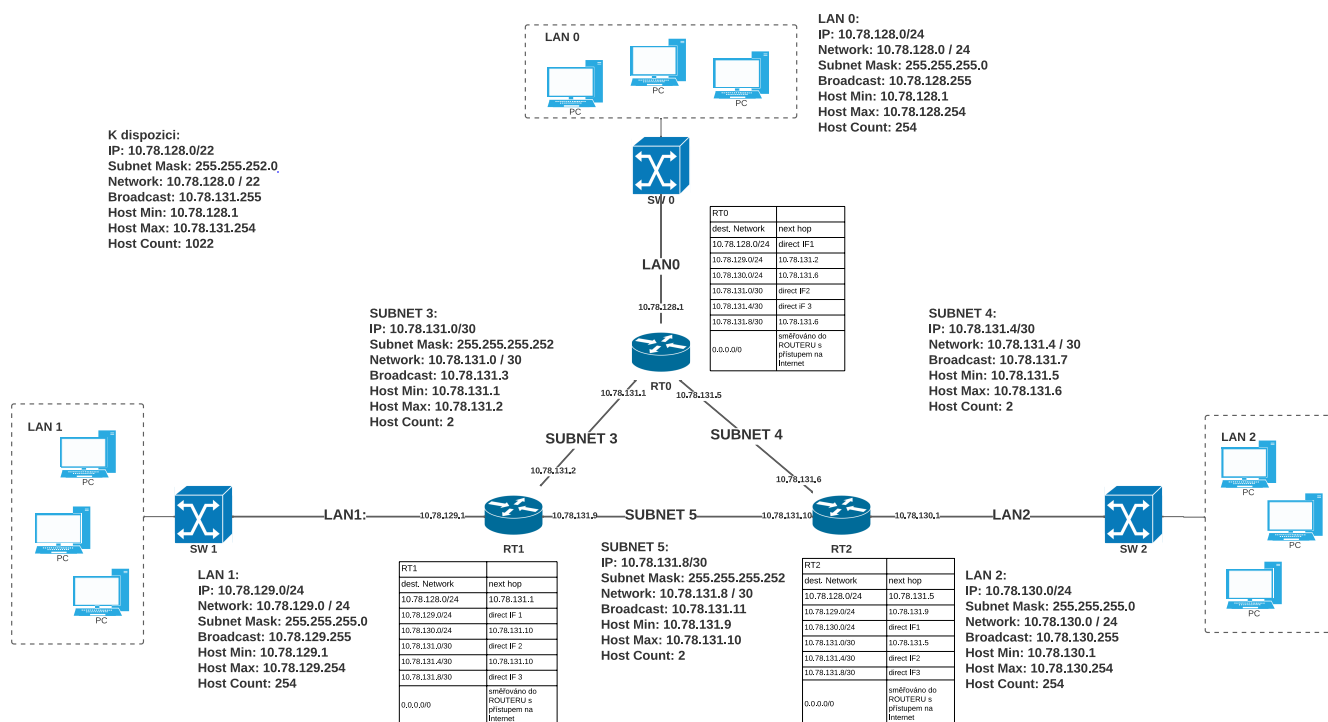


Návrh sítě:

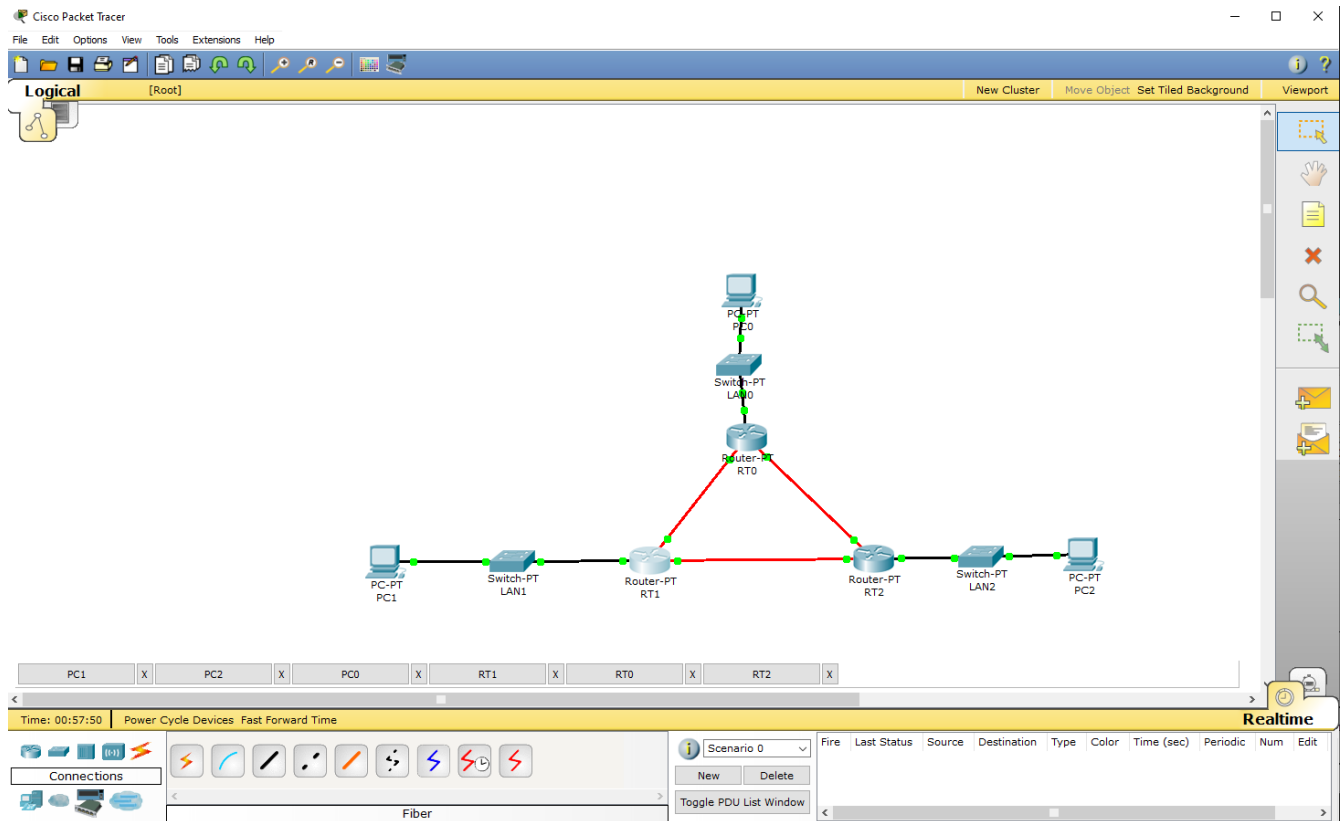
Vygenerovaná síť má následující adresní prostor:

ID:	211538	Name:	Filip Paul
Address space:			
10.78.128.0/22			
Divide your IP address space into subnetworks as follows: a) make the address space in LAN0-LAN2 subnets as large as possible, b) make the address space in subnets connecting the routers as small as possible. Follow the convention that the gateway of the local networks (LAN0-LAN2) has the lowest possible IP address of the range. (Note: After this division there will reserve for another subnets which could connect only two devices)			

Bohužel jsem se podíval na obrázek a rovnou jsem začal vytvářet síť podle postupu, který jsme dělali v prezenčních cvičích, což tak nějak neodpovídá zadání. Především tomu, že každá podsíť může využít až 31 IP adres. V mojem řešení je pro routery propojené napřímo dedikovaná menší síť. Celkově vznikne 6 podsítí. Pro výpočty parametrů sítě jsem využil vlastní python script, který si můžete zobrazit buď v mém GITHUB repozitáři [zde](#) nebo na konci dokumentu. V github repozitáři taky naleznete [cisco-project.pkt](#) soubor z PC simulace.



PC simulace:



PC0
Command Prompt

```

Reply from 10.78.131.5: bytes=32 time=70ms TTL=254
Reply from 10.78.131.2: bytes=32 time=60ms TTL=254
Reply from 10.78.131.2: bytes=32 time=52ms TTL=254
Reply from 10.78.131.2: bytes=32 time=40ms TTL=254

Ping statistics for 10.78.131.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 40ms, Maximum = 70ms, Average = 55ms

PC>ping 10.78.129.2

Pinging 10.78.129.2 with 32 bytes of data:
Reply from 10.78.129.2: bytes=32 time=60ms TTL=126
Reply from 10.78.129.2: bytes=32 time=100ms TTL=126
Reply from 10.78.129.2: bytes=32 time=100ms TTL=126
Reply from 10.78.129.2: bytes=32 time=100ms TTL=126

Ping statistics for 10.78.129.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 90ms, Maximum = 100ms, Average = 97ms

PC>

```

RT0
Static Routes

Network	Mask	Next Hop
10.78.131.8	255.255.255.252	10.78.131.2

Remove

Equivalent IOS Commands

```

Router(config)#ip route 10.78.129.0 255.255.255.0 10.78.131.2
Router(config)#ip route 10.78.130.0 255.255.255.0 10.78.131.2
Router(config)#

```

PC1
Command Prompt

```

Reply from 10.78.131.5: bytes=32 time=70ms TTL=254
Reply from 10.78.131.8: bytes=32 time=70ms TTL=254
Reply from 10.78.131.5: bytes=32 time=70ms TTL=254
Reply from 10.78.131.5: bytes=32 time=70ms TTL=254

Ping statistics for 10.78.131.5:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 40ms, Maximum = 70ms, Average = 67ms

PC>ping 10.78.130.2

Pinging 10.78.130.2 with 32 bytes of data:
Reply from 10.78.130.2: bytes=32 time=100ms TTL=126
Reply from 10.78.130.2: bytes=32 time=100ms TTL=126
Reply from 10.78.130.2: bytes=32 time=100ms TTL=126
Reply from 10.78.130.2: bytes=32 time=100ms TTL=126

Ping statistics for 10.78.130.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 80ms, Maximum = 100ms, Average = 90ms

PC>

```

RT1
Static Routes

Network	Mask	Next Hop
10.78.131.4	255.255.255.252	10.78.131.10

Remove

Equivalent IOS Commands

```

Router(config)#ip route 10.78.131.4 255.255.255.252 10.78.131.10
Router(config)#

```

PC2
Command Prompt

```

Reply from 10.78.131.5: bytes=32 time=60ms TTL=254
Reply from 10.78.131.2: bytes=32 time=70ms TTL=254
Reply from 10.78.131.2: bytes=32 time=70ms TTL=254
Reply from 10.78.131.2: bytes=32 time=70ms TTL=254

Ping statistics for 10.78.131.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 60ms, Maximum = 70ms, Average = 67ms

PC>ping 10.78.131.2

Pinging 10.78.131.2 with 32 bytes of data:
Reply from 10.78.131.2: bytes=32 time=62ms TTL=254
Reply from 10.78.131.2: bytes=32 time=70ms TTL=254
Reply from 10.78.131.2: bytes=32 time=70ms TTL=254
Reply from 10.78.131.2: bytes=32 time=100ms TTL=254

Ping statistics for 10.78.131.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 60ms, Maximum = 110ms, Average = 78ms

PC>

```

RT2
Static Routes

Network	Mask	Next Hop
10.78.131.0	255.255.255.252	10.78.131.5

Remove

Equivalent IOS Commands

```

Router(config)#ip route 10.78.131.0 255.255.255.252 10.78.131.5
Router(config)#ip route 10.78.131.0 255.255.255.252 10.78.131.5
Router(config)#ip route 10.78.131.0 255.255.255.252 10.78.131.5
Router(config)#

```

Přílohy:

calc.py

```
1 from myIPAddress import myIPAddress #import my custom class
2
3 print("\nK dispozici:")
4 myIPAddress("10.78.128.0/22").printInfo()
5
6 print("\nLAN 0")
7 myIPAddress("10.78.128.0/24").printInfo()
8
9 print("\nLAN 1")
10 myIPAddress("10.78.129.0/24").printInfo()
11
12 print("\nLAN 2")
13 myIPAddress("10.78.130.0/24").printInfo()
14
15 print("\nSUBNET 3")
16 myIPAddress("10.78.131.0/30").printInfo()
17
18 print("\nSUBNET 4")
19 myIPAddress("10.78.131.4/30").printInfo()
20
21 print("\nSUBNET 5")
22 myIPAddress("10.78.131.8/30").printInfo()
```

OUTPUT: K dispozici:

IP Address full format: 10.78.128.0/22

IP: 10.78.128.0

IP Bits: 00001010.01001110.10000000.00000000

Subnet Mask: 255.255.252.0 suffix: 22

Subnet Mask Bits: 11111111.11111111.11111100.00000000

Network: 10.78.128.0 / 22

Broadcast: 10.78.131.255

Host Min: 10.78.128.1

Host Max: 10.78.131.254

Host Count: 1022

LAN 0

IP Address full format: 10.78.128.0/24

IP: 10.78.128.0

IP Bits: 00001010.01001110.10000000.00000000

Subnet Mask: 255.255.255.0 suffix: 24

Subnet Mask Bits: 11111111.11111111.11111111.00000000

Network: 10.78.128.0 / 24

Broadcast: 10.78.128.255

Host Min: 10.78.128.1

Host Max: 10.78.128.254

Host Count: 254

LAN 1

IP Address full format: 10.78.129.0/24

IP: 10.78.129.0

IP Bits: 00001010.01001110.10000001.00000000

Subnet Mask: 255.255.255.0 suffix: 24

Subnet Mask Bits: 11111111.11111111.11111111.00000000

Network: 10.78.129.0 / 24

Broadcast: 10.78.129.255

Host Min: 10.78.129.1

Host Max: 10.78.129.254

Host Count: 254

LAN 2

IP Address full format: 10.78.130.0/24
IP: 10.78.130.0
IP Bits: 00001010.01001110.10000010.00000000
Subnet Mask: 255.255.255.0 suffix: 24
Subnet Mask Bits: 11111111.11111111.11111111.00000000

Network: 10.78.130.0 / 24
Broadcast: 10.78.130.255
Host Min: 10.78.130.1
Host Max: 10.78.130.254
Host Count: 254

SUBNET 3

IP Address full format: 10.78.131.0/30
IP: 10.78.131.0
IP Bits: 00001010.01001110.10000011.00000000
Subnet Mask: 255.255.255.252 suffix: 30
Subnet Mask Bits: 11111111.11111111.11111111.11111100

Network: 10.78.131.0 / 30
Broadcast: 10.78.131.3
Host Min: 10.78.131.1
Host Max: 10.78.131.2
Host Count: 2

SUBNET 4

IP Address full format: 10.78.131.4/30
IP: 10.78.131.4
IP Bits: 00001010.01001110.10000011.00000100
Subnet Mask: 255.255.255.252 suffix: 30
Subnet Mask Bits: 11111111.11111111.11111111.11111100

Network: 10.78.131.4 / 30
Broadcast: 10.78.131.7
Host Min: 10.78.131.5
Host Max: 10.78.131.6
Host Count: 2

SUBNET 5

IP Address full format: 10.78.131.8/30
IP: 10.78.131.8
IP Bits: 00001010.01001110.10000011.00001000
Subnet Mask: 255.255.255.252 suffix: 30
Subnet Mask Bits: 11111111.11111111.11111111.11111100

Network: 10.78.131.8 / 30
Broadcast: 10.78.131.11
Host Min: 10.78.131.9
Host Max: 10.78.131.10
Host Count: 2

myIPAddress.py

```
1 class myIPAddress():
2     def __init__(self, ipaddr:str):
3         self.ipaddr = ipaddr
4         self.IP = ipaddr[:ipaddr.find("/")]
5         self.IP_bits = self.get_IP_bits(ipaddr)
6         self.subnet_mask, self.subnet_mask_bits, self.suffix = self.get_subnet_mask(ipaddr)
7         self.broadcast = ""
8         self.network = ""
9         self.host_min = ""
10        self.host_max = ""
11        self.host_count = ""
12        self.get_IP_range()
13
14    def get_subnet_mask(self, ipaddr:str):
15        pos = ipaddr.find("/")
16        subnet = "1"* int(ipaddr[pos+1:]) + "0"*(32-int(ipaddr[pos+1:]))
17        mask = ""
18        mask_bits = ""
19        for Bytes in range(0,31,8):
20            mask += f"{int(subnet[ Bytes: Bytes+8], 2)}."
21            mask_bits += f"{subnet[ Bytes: Bytes+8]}."
22
23        return mask[:-1], mask_bits[:-1], int(ipaddr[pos+1:]) #remove last dot
24
25    def get_IP_bits(self, ipaddr:str):
26        ipaddr = ipaddr[:ipaddr.find("/")]
27
28        IP_bits = ""
29        for Bytes in ipaddr.split("."):
30            IP_bits += f"{bin(int(Bytes))[2:].zfill(8)}."
31        return IP_bits[:-1] #remove last dot
32
33    def get_IP_range(self):
34        current_IP_bytes = self.IP_bits.split(".")
35        current_SUBNET_bytes = self.subnet_mask_bits.split(".")
36
37        for IP_Byte, MASK_Byte in zip(current_IP_bytes, current_SUBNET_bytes):
38            i = 0
39            network_substract = 0
40            broadcast_to_add = 0
41            for IP_BIT, MASK_BIT in zip(IP_Byte, MASK_Byte):
42                if MASK_BIT == "0":
43                    broadcast_to_add += 2**(7-i)
44                if MASK_BIT == "0" and IP_BIT == "1":
45                    network_substract += 2**(7-i)
46                i+=1
47
48            self.broadcast += f"{int(IP_Byte,2)+broadcast_to_add-network_substract}."
49            self.network += f"{int(IP_Byte,2) - network_substract}."
50
51
52
53        self.broadcast = self.broadcast[:-1]
54        self.network = self.network[:-1]
55        last_byte_pos= self.broadcast.rfind(".")
56        self.host_max = self.broadcast[:last_byte_pos+1] + f"{int(self.broadcast.split('.')[-1])
-1}"
57
58        last_byte_pos= self.IP.rfind(".")
59        self.host_min = self.network[:last_byte_pos+1] + f"{int(self.network.split('.')[-1])+1}"
60        self.host_count = 2**(32-self.suffix) - 2
61
62    def printInfo(self):
63        print("IP Address full format:", self.ipaddr)
64        print("IP:", self.IP)
65        print("IP Bits:", self.IP_bits)
66        print("Subnet Mask:", self.subnet_mask, "suffix:", self.suffix)
67        print("Subnet Mask Bits:", self.subnet_mask_bits)
68        print("\nNetwork:", self.network, "/", self.suffix)
69        print("Broadcast:", self.broadcast)
70        print("Host Min:", self.host_min)
71        print("Host Max:", self.host_max)
72        print("Host Count:", self.host_count)
```