Лабораторная работа №8

Кущ Артем, БПМ-18-1

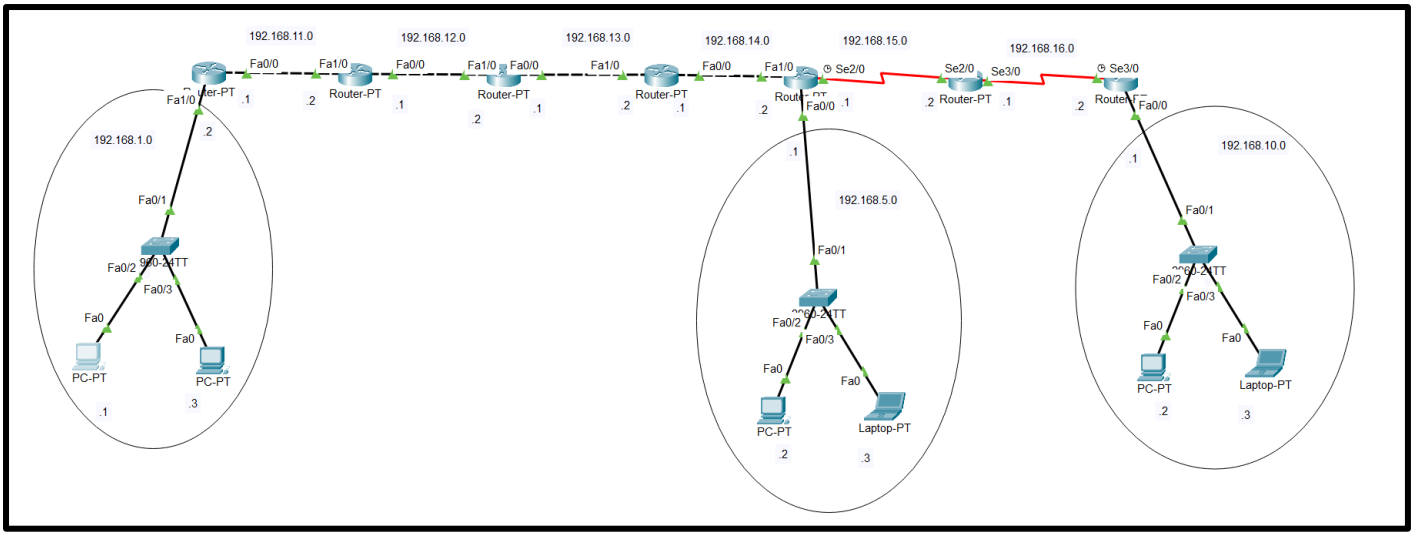
1) Собрал схему
2) Задал ip и шлюзы по умолчанию
3) Настроил роутеры с помощью скрипта.
4) Пинги идут.
5) Настроил vty на 192.168.1.2-роутере
6) Настроил ACL (стандартный, дабы фильтровать только адрес отправителя), с помощью скрипта
7) Применил ACL на vty
8) >sh access-lists

Показывает разрешённые IP и количество запросов с них.

9) Если кроме "permit" дать команду "deny", то мы сможем определять поступающие запросы с запрещённых IP.
10) Если при создании ACL указать неправильную маску (например, вместо 255→254), то могут пропускаться некоторые IP.

В случае маски ....254 – чётные/нечётные адреса

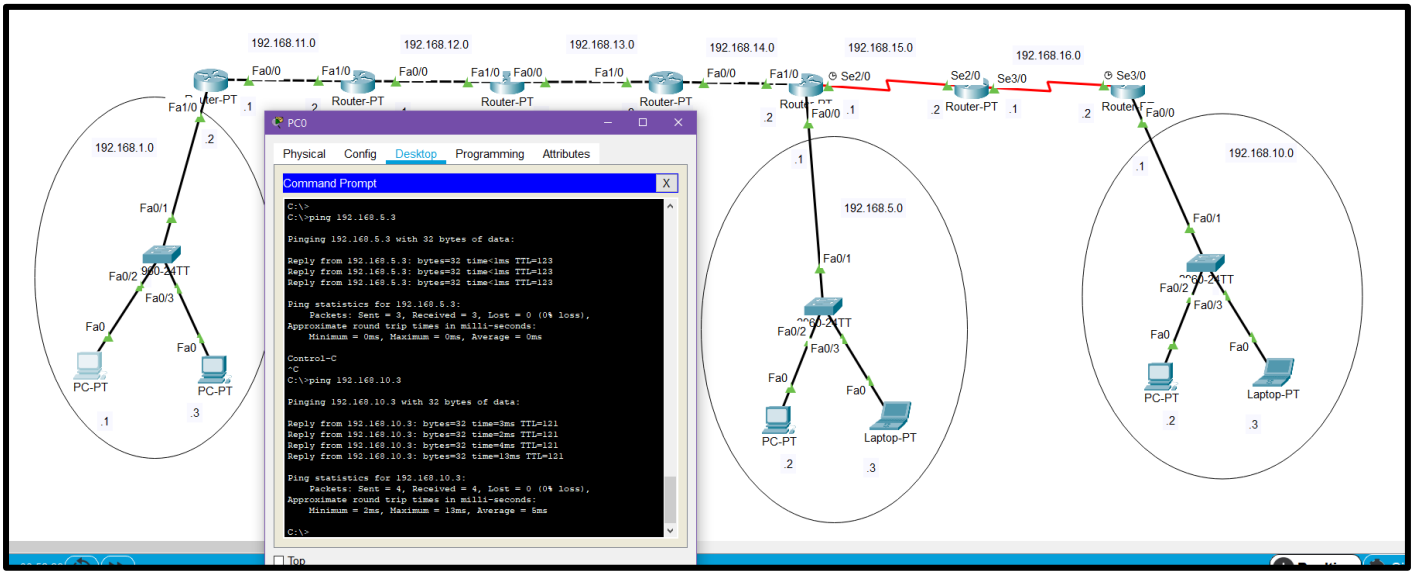
11) "Extended" позволяет запретить только определённый тип трафика, определить источники и место назначения.
12) Важно! Расширенные ACL ставятся ближе к отправителю, а стандартные – ближе к получателю
13) Для настройки Nat можно либо настроить OSPF, либо loopback-интерфейс, либо прописать вручную.
14) Мне понравилось через loopback-интерфейс. Удобно.
15) Важно! Nat работает "в одну сторону". Нельзя обратиться от других сетей по адресу, который маскирует Nat.



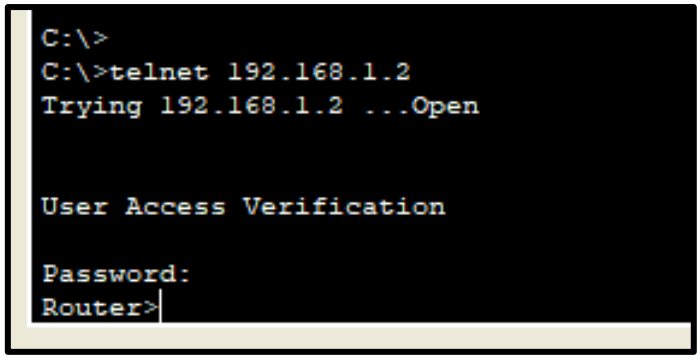
3) Настройка роутеров с помощью скрипта, написанном на языке Python:

```
networks = [1, 11, 12, 13, 14, 5]
for i in range(len(networks) - 1):
    print(f'Router №{i + 1}:')
    print('en\nconf t')
    print(f'int fa0/0')
    print(f'ip address 192.168.{networks[i + 1]}.1 255.255.255.0')
    print(f'no shutdown')
    print(f'int fa1/0')
    print(f'ip address 192.168.{networks[i]}.2 255.255.255.0')
    print(f'no shutdown')
    print('exit\nrouter eigrp 1')
    print(f'network 192.168.{networks[i + 1]}.0 0.0.0.255')
    print(f'network 192.168.{networks[i]}.0 0.0.0.255')
    print('exit\nexit\n\n')
```

4) Проверка на ***ping*** после настройки:



5) Настроил VTY на роутере и успешно подключился по telnet:



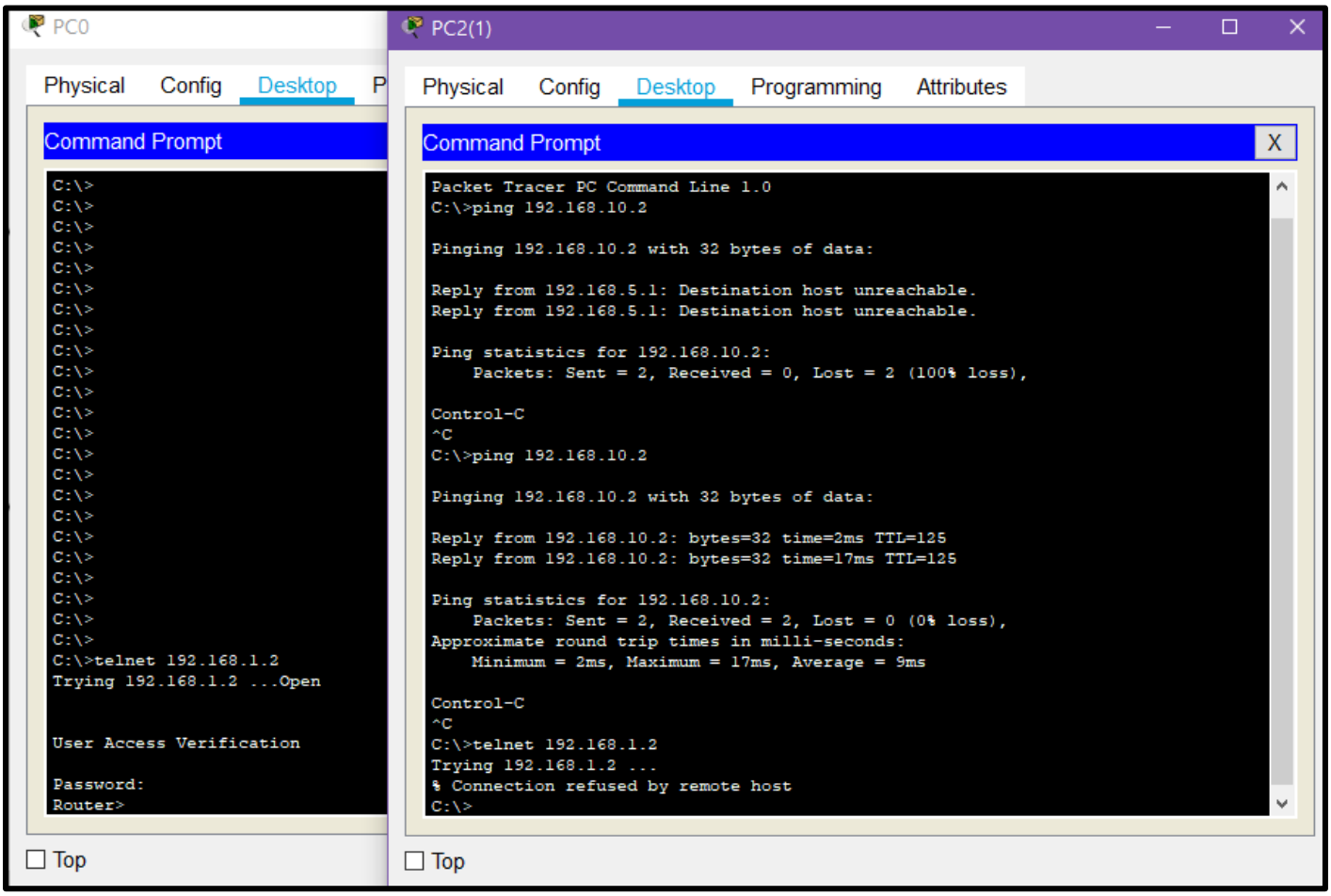
```
en
conf t
ip access-list standard ACL1
permit 192.168.1.0 0.0.0.255
deny 192.168.5.0 0.0.0.255
exit
line vty 0 15
access-class ACL1 in
exit
```

7) Провел настройку ACL. Разрешил удаленный доступ устройства из сети 192.168.1.0 и применил на виртуальное подключение для входящих запросов.

```
Router#sh access-lists
Standard IP access list ACL1
    10 permit 192.168.1.0 0.0.0.255 (2 match(es))
    20 deny 192.168.10.0 0.0.0.255 (16 match(es))

Router#
```

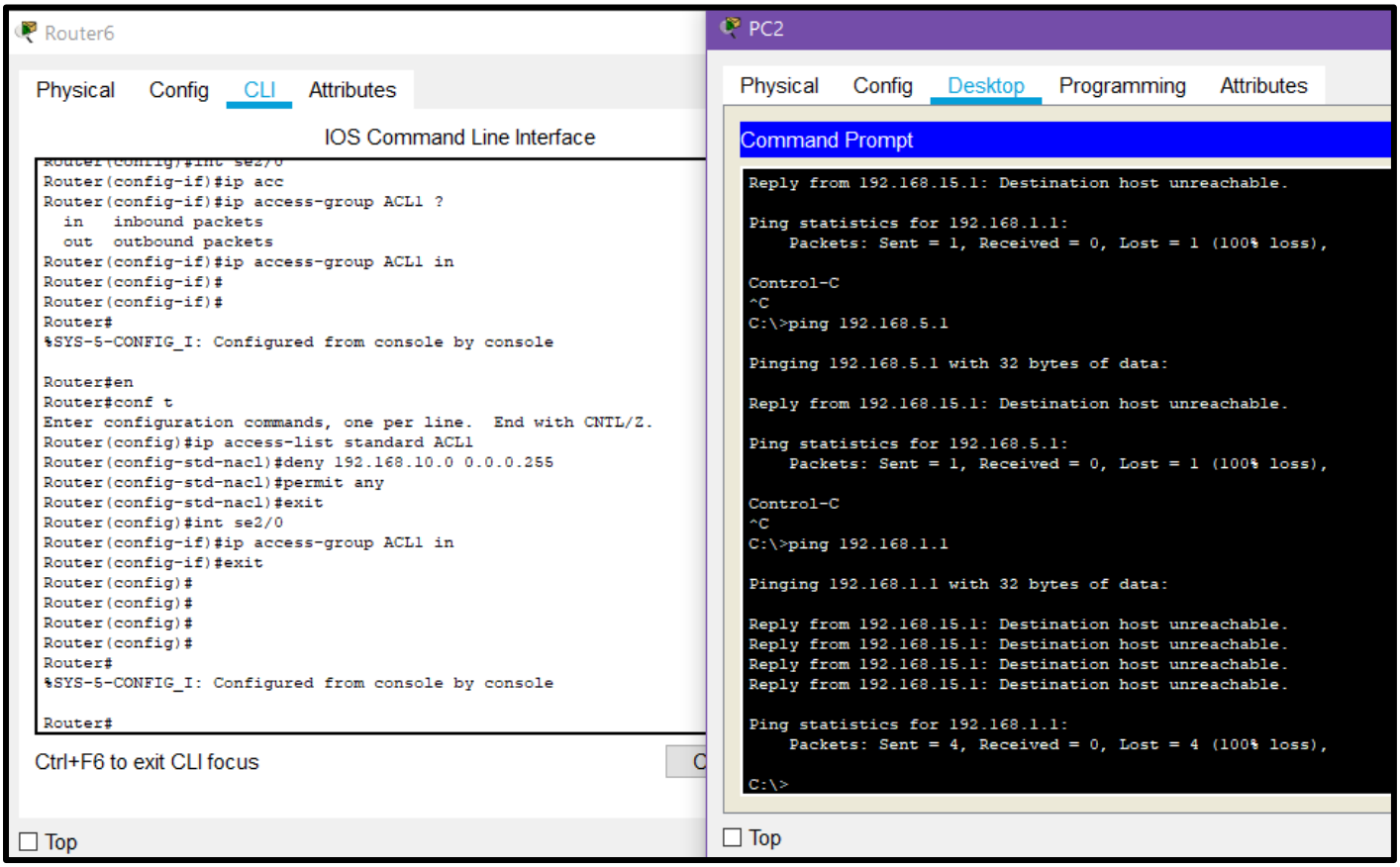
После настройки ACL все устройства из сети 192.168.1.0 могут получить удаленный доступ к роутеру; остальные не могут:



Системный администратор, работающий в крупнейшей компании, узнал, что непорядочный студент-стажер некорректно настроил сеть на несколько стационарных ПК с адресом 192.168.10.0: не отключил свободные порты, оставил автоматические режимы интерфейсов, включил DHCP и CDP, забыл установить пароли на свитчи, не включил port-security и т. д.
В связи с халатностью работника, в сеть проник злоумышленник: начал засорять таблицу DHCP, прослушивать трафик, отправлять “плохие” пакеты посредством двойного тегирования и удаленно подключаться к коммутаторам. Нужно срочно предпринимать какие-то действия: возможности отключить сеть физически нет, поэтому быстро настроим ACL на “опасный” порт и запретим пересылку пакетов из этой сети.

```
en
conf t
ip access-list standard ACL1
deny 192.168.10.0 0.0.0.255
permit any
exit
int se2/0
ip access-group ACL1 in
exit
```

Применим данный скрипт на “среднем роутере”. Успех, пакеты в другие сети не идет.

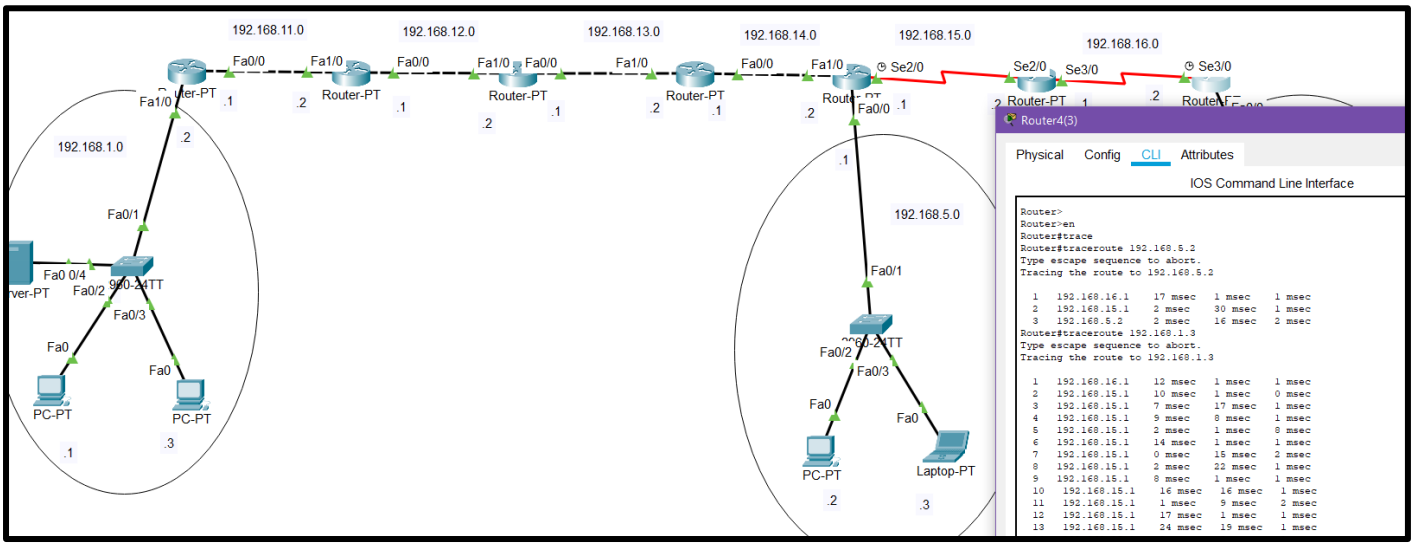


К сожалению, предыдущая атака мошенника не прошла бесследно… Нападению подверглась и сеть 192.168.16.0: с неё начали поступать “тяжелые” UDP пакеты. С помощью расширенного доступа ACL защитим сеть 192.168.1.0 от пакетов такого типа:

```
en
conf t
ip access-list standard ACLE2
deny udp 192.168.16.0 0.0.0.255 192.168.1.0 0.0.0.255
permit ip any any
exit
int se2/0
ip access-group ACLE2 in
exit
```

Успешно. UDP пакеты больше не идут в сеть 192.168.1.0, но продолжают идти в другие. **Что интересно, при попытке отослать UDP пакеты в сеть 192.168.1.0 произошло зацикливание.**

- **Почему?**



Настройка NAT на верхнем левом роутере:

```
en
conf t
ip access-list standard ACL1
permit 192.168.1.0 0.0.0.255
exit
ip nat pool P100 100.0.0.1 100.255.255.255 netmask 255.0.0.0
ip nat inside source list ACL1 pool P100
int f0/0
ip nat outside
int f1/0
ip nat inside
```

- **Вопрос: 100.255.255.255 или 100.255.255.254?**

```
en
conf t
int loopback 100
ip address 100.100.100.1 255.0.0.0
exit
router eigrp 1
network 100.0.0.0 0.255.255.255
exit
```

- NAT переводит внутренние локальные адреса во внутренние глобальные адреса, аналогично, PAT преобразует частные незарегистрированные IP-адреса в общедоступные зарегистрированные IP-адреса, но в отличие от NAT он также использует номера портов источника, и нескольким хостам может быть назначен один и тот же IP, имеющий разные номера портов.
- PAT является формой динамического NAT.
- NAT использует IP-адреса в процессе трансляции, тогда как PAT использует IP-адреса вместе с номерами портов.

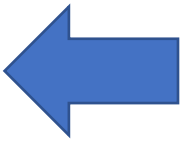
```
C:\>tracert 192.168.1.1

Tracing route to 192.168.1.1 over a maximum of 30 hops:

  1    0 ms      1 ms      0 ms      192.168.5.1
  2    0 ms      0 ms      0 ms      192.168.14.1
  3    1 ms      0 ms      0 ms      192.168.13.1
  4    0 ms      1 ms      1 ms      192.168.12.1
  5    1 ms      0 ms      0 ms      192.168.11.1
  6    0 ms      0 ms      10 ms     100.0.0.2

Trace complete.

C:\>
```



NAT

```
en
conf t
ip access-list standard ACL1
permit 192.168.1.0 0.0.0.255
exit
ip nat inside source list ACL1 int f0/0 overload
int f0/0
ip nat outside
int f1/0
ip nat inside
```

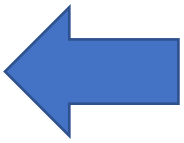
```
C:\>tracert 192.168.1.1

Tracing route to 192.168.1.1 over a maximum of 30 hops:

  1    1 ms      1 ms      0 ms      192.168.5.1
  2    1 ms      0 ms      0 ms      192.168.14.1
  3    0 ms      0 ms      0 ms      192.168.13.1
  4    1 ms      0 ms      0 ms      192.168.12.1
  5    0 ms      0 ms      0 ms      192.168.11.1
  6    0 ms      0 ms      0 ms      192.168.11.1

Trace complete.

C:\>
```



PAT