

Q1 SNMP (Section 9.9)

1 Point

Q1.1 SNMP community string (Section 9.9, Exercise 2)

0.5 Points

What is the difference in the output when running `snmpwalk` on the "server" host with "public" as the community string, versus "secret" as the community string, **and why**? Show output in each case, and explain.

Running `snmpwalk` on the server host with "secret" as the community string will output much more information than using "public" as the community string. Since it made a small amount of management information available to the public but much more information available to the designated host that we want to be able to monitor the network condition.

▼ server_public.png

 Download

```

ty2069@server:~$ snmpwalk -v 2c -c public router-int
iso.3.6.1.2.1.1.1.0 = STRING: "Linux router-int.lab9-ty2069.ch-geni-net.instageni.ucsd.edu 4.15.0-121-generic #123-Ubuntu SMP Mon Oct 5 16:16:40 UTC 2020 x86_64"
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.8072.3.2.10
iso.3.6.1.2.1.1.3.0 = Timeticks: (1028) 0:00:10.28
iso.3.6.1.2.1.1.4.0 = STRING: "Me <me@example.org>"
iso.3.6.1.2.1.1.5.0 = STRING: "router-int.lab9-ty2069.ch-geni-net.instageni.ucsd.edu"
iso.3.6.1.2.1.1.6.0 = STRING: "Sitting on the Dock of the Bay"
iso.3.6.1.2.1.1.7.0 = INTEGER: 72
iso.3.6.1.2.1.1.8.0 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.2.1 = OID: iso.3.6.1.6.3.11.3.1.1
iso.3.6.1.2.1.1.9.1.2.2 = OID: iso.3.6.1.6.3.15.2.1.1
iso.3.6.1.2.1.1.9.1.2.3 = OID: iso.3.6.1.6.3.10.3.1.1
iso.3.6.1.2.1.1.9.1.2.4 = OID: iso.3.6.1.6.3.1
iso.3.6.1.2.1.1.9.1.2.5 = OID: iso.3.6.1.6.3.16.2.2.1
iso.3.6.1.2.1.1.9.1.2.6 = OID: iso.3.6.1.2.1.49
iso.3.6.1.2.1.1.9.1.2.7 = OID: iso.3.6.1.2.1.4
iso.3.6.1.2.1.1.9.1.2.8 = OID: iso.3.6.1.2.1.50
iso.3.6.1.2.1.1.9.1.2.9 = OID: iso.3.6.1.6.3.13.3.1.3
iso.3.6.1.2.1.1.9.1.2.10 = OID: iso.3.6.1.2.1.92
iso.3.6.1.2.1.1.9.1.3.1 = STRING: "The MIB for Message Processing and Dispatching."
iso.3.6.1.2.1.1.9.1.3.2 = STRING: "The management information definitions for the SNMP User-based Security Model."
iso.3.6.1.2.1.1.9.1.3.3 = STRING: "The SNMP Management Architecture MIB."
iso.3.6.1.2.1.1.9.1.3.4 = STRING: "The MIB module for SNMPv2 entities"
iso.3.6.1.2.1.1.9.1.3.5 = STRING: "View-based Access Control Model for SNMP."
iso.3.6.1.2.1.1.9.1.3.6 = STRING: "The MIB module for managing TCP implementations"
iso.3.6.1.2.1.1.9.1.3.7 = STRING: "The MIB module for managing IP and ICMP implementations"
iso.3.6.1.2.1.1.9.1.3.8 = STRING: "The MIB module for managing UDP implementations"
iso.3.6.1.2.1.1.9.1.3.9 = STRING: "The MIB modules for managing SNMP Notification, plus filtering."
iso.3.6.1.2.1.1.9.1.3.10 = STRING: "The MIB module for logging SNMP Notifications."
iso.3.6.1.2.1.1.9.1.4.1 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.2 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.3 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.4 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.5 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.6 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.7 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.8 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.9 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.10 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.25.1.1.0 = Timeticks: (16948729) 1 day, 23:04:47.29
iso.3.6.1.2.1.25.1.2.0 = Hex-STRING: 07 E4 0C 04 14 1A 17 00 2D 05 00
iso.3.6.1.2.1.25.1.3.0 = INTEGER: 393216
iso.3.6.1.2.1.25.1.4.0 = STRING: "root=UUID=5fac4c69-51f6-47af-9773-a0d722426942 ro console=ttyS0,115200 root=/dev/xvda1 ro console=hvc0 xencons=tty apparmor=0 se"
iso.3.6.1.2.1.25.1.5.0 = Gauge32: 1
iso.3.6.1.2.1.25.1.6.0 = Gauge32: 86
iso.3.6.1.2.1.25.1.7.0 = INTEGER: 0
iso.3.6.1.2.1.25.1.7.0 = No more variables left in this MIB View (It is past the end of the MIB tree)

```

▼ server_secret.png

Download

```

ty2069@server:~$ snmpwalk -v 2c -c secret router-int
iso.3.6.1.2.1.1.1.0 = STRING: "Linux router-int.lab9-ty2069.ch-geni-net.instagen
i.ucsd.edu 4.15.0-121-generic #123-Ubuntu SMP Mon Oct 5 16:16:40 UTC 2020 x86_64
"
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.8072.3.2.10
iso.3.6.1.2.1.1.3.0 = Timeticks: (1061) 0:00:10.61
iso.3.6.1.2.1.1.4.0 = STRING: "Me <me@example.org>"
iso.3.6.1.2.1.1.5.0 = STRING: "router-int.lab9-ty2069.ch-geni-net.instageni.ucsd
.edu"
iso.3.6.1.2.1.1.6.0 = STRING: "Sitting on the Dock of the Bay"
iso.3.6.1.2.1.1.7.0 = INTEGER: 72
iso.3.6.1.2.1.1.8.0 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.2.1 = OID: iso.3.6.1.6.3.11.3.1.1
iso.3.6.1.2.1.1.9.1.2.2 = OID: iso.3.6.1.6.3.15.2.1.1
iso.3.6.1.2.1.1.9.1.2.3 = OID: iso.3.6.1.6.3.10.3.1.1
iso.3.6.1.2.1.1.9.1.2.4 = OID: iso.3.6.1.6.3.1
iso.3.6.1.2.1.1.9.1.2.5 = OID: iso.3.6.1.6.3.16.2.2.1
iso.3.6.1.2.1.1.9.1.2.6 = OID: iso.3.6.1.2.1.49
iso.3.6.1.2.1.1.9.1.2.7 = OID: iso.3.6.1.2.1.4
iso.3.6.1.2.1.1.9.1.2.8 = OID: iso.3.6.1.2.1.50
iso.3.6.1.2.1.1.9.1.2.9 = OID: iso.3.6.1.6.3.13.3.1.3
iso.3.6.1.2.1.1.9.1.2.10 = OID: iso.3.6.1.2.1.92
iso.3.6.1.2.1.1.9.1.3.1 = STRING: "The MIB for Message Processing and Dispatchin
g."
iso.3.6.1.2.1.1.9.1.3.2 = STRING: "The management information definitions for th
e SNMP User-based Security Model."
iso.3.6.1.2.1.1.9.1.3.3 = STRING: "The SNMP Management Architecture MIB."
iso.3.6.1.2.1.1.9.1.3.4 = STRING: "The MIB module for SNMPv2 entities"
iso.3.6.1.2.1.1.9.1.3.5 = STRING: "View-based Access Control Model for SNMP."
iso.3.6.1.2.1.1.9.1.3.6 = STRING: "The MIB module for managing TCP implementatio
ns"
iso.3.6.1.2.1.1.9.1.3.7 = STRING: "The MIB module for managing IP and ICMP imple
mentations"
iso.3.6.1.2.1.1.9.1.3.8 = STRING: "The MIB module for managing UDP implementatio
ns"
iso.3.6.1.2.1.1.9.1.3.9 = STRING: "The MIB modules for managing SNMP Notificatio
n, plus filtering."
iso.3.6.1.2.1.1.9.1.3.10 = STRING: "The MIB module for logging SNMP Notification
s."
iso.3.6.1.2.1.1.9.1.4.1 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.2 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.3 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.4 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.5 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.6 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.7 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.8 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.9 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.10 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.2.1.0 = INTEGER: 5
iso.3.6.1.2.1.2.2.1.1.1 = INTEGER: 1
iso.3.6.1.2.1.2.2.1.1.2 = INTEGER: 2
iso.3.6.1.2.1.2.2.1.1.3 = INTEGER: 3
iso.3.6.1.2.1.2.2.1.1.4 = INTEGER: 4
iso.3.6.1.2.1.2.2.1.1.5 = INTEGER: 5
iso.3.6.1.2.1.2.2.1.2.1 = STRING: "lo"
iso.3.6.1.2.1.2.2.1.2.2 = STRING: "eth0"
iso.3.6.1.2.1.2.2.1.2.3 = STRING: "eth1"
iso.3.6.1.2.1.2.2.1.2.4 = STRING: "eth2"
iso.3.6.1.2.1.2.2.1.2.5 = STRING: "eth3"
iso.3.6.1.2.1.2.2.1.3.1 = INTEGER: 24
iso.3.6.1.2.1.2.2.1.3.2 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.3 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.4 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.5 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.4.1 = INTEGER: 65536
iso.3.6.1.2.1.2.2.1.4.2 = INTEGER: 1500

```

What is the difference in the output when running `snmpwalk` with "secret" as the community string on the "server" host, versus with "secret" as the community string on the "romeo" host, **and why**? Show output in each case, and explain.

On romeo host, we will have No Response as output since in the router's snmpd configuration file we set up the access control rule that only allows the server host could access the secret information.

▼ romeo_secret.png

Download

```
ty2069@romeo:~$ snmpwalk -v 2c -c secret router-int
Timeout: No Response from router-int
```

▼ server_secret.png

Download

```
ty2069@server:~$ snmpwalk -v 2c -c secret router-int
iso.3.6.1.2.1.1.1.0 = STRING: "Linux router-int.lab9-ty2069.ch-geni-net.instagen
i.ucsd.edu 4.15.0-121-generic #123-Ubuntu SMP Mon Oct 5 16:16:40 UTC 2020 x86_64
"
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.8072.3.2.10
iso.3.6.1.2.1.1.3.0 = Timeticks: (1061) 0:00:10.61
iso.3.6.1.2.1.1.4.0 = STRING: "Me <me@example.org>"
iso.3.6.1.2.1.1.5.0 = STRING: "router-int.lab9-ty2069.ch-geni-net.instageni.ucsd
.edu"
iso.3.6.1.2.1.1.6.0 = STRING: "Sitting on the Dock of the Bay"
iso.3.6.1.2.1.1.7.0 = INTEGER: 72
iso.3.6.1.2.1.1.8.0 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.2.1 = OID: iso.3.6.1.6.3.11.3.1.1
iso.3.6.1.2.1.1.9.1.2.2 = OID: iso.3.6.1.6.3.15.2.1.1
iso.3.6.1.2.1.1.9.1.2.3 = OID: iso.3.6.1.6.3.10.3.1.1
iso.3.6.1.2.1.1.9.1.2.4 = OID: iso.3.6.1.6.3.1
iso.3.6.1.2.1.1.9.1.2.5 = OID: iso.3.6.1.6.3.16.2.2.1
iso.3.6.1.2.1.1.9.1.2.6 = OID: iso.3.6.1.2.1.49
iso.3.6.1.2.1.1.9.1.2.7 = OID: iso.3.6.1.2.1.4
iso.3.6.1.2.1.1.9.1.2.8 = OID: iso.3.6.1.2.1.50
iso.3.6.1.2.1.1.9.1.2.9 = OID: iso.3.6.1.6.3.13.3.1.3
iso.3.6.1.2.1.1.9.1.2.10 = OID: iso.3.6.1.2.1.92
iso.3.6.1.2.1.1.9.1.3.1 = STRING: "The MIB for Message Processing and Dispatchin
g."
iso.3.6.1.2.1.1.9.1.3.2 = STRING: "The management information definitions for th
e SNMP User-based Security Model."
iso.3.6.1.2.1.1.9.1.3.3 = STRING: "The SNMP Management Architecture MIB."
iso.3.6.1.2.1.1.9.1.3.4 = STRING: "The MIB module for SNMPv2 entities"
iso.3.6.1.2.1.1.9.1.3.5 = STRING: "View-based Access Control Model for SNMP."
iso.3.6.1.2.1.1.9.1.3.6 = STRING: "The MIB module for managing TCP implementatio
ns"
iso.3.6.1.2.1.1.9.1.3.7 = STRING: "The MIB module for managing IP and ICMP imple
mentations"
iso.3.6.1.2.1.1.9.1.3.8 = STRING: "The MIB module for managing UDP implementatio
ns"
iso.3.6.1.2.1.1.9.1.3.9 = STRING: "The MIB modules for managing SNMP Notificatio
n, plus filtering."
iso.3.6.1.2.1.1.9.1.3.10 = STRING: "The MIB module for logging SNMP Notification
s."
iso.3.6.1.2.1.1.9.1.4.1 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.2 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.3 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.4 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.5 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.6 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.7 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.8 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.9 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.10 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.2.1.0 = INTEGER: 5
iso.3.6.1.2.1.2.2.1.1.1 = INTEGER: 1
iso.3.6.1.2.1.2.2.1.1.2 = INTEGER: 2
iso.3.6.1.2.1.2.2.1.1.3 = INTEGER: 3
iso.3.6.1.2.1.2.2.1.1.4 = INTEGER: 4
iso.3.6.1.2.1.2.2.1.1.5 = INTEGER: 5
iso.3.6.1.2.1.2.2.1.2.1 = STRING: "lo"
iso.3.6.1.2.1.2.2.1.2.2 = STRING: "eth0"
iso.3.6.1.2.1.2.2.1.2.3 = STRING: "eth1"
iso.3.6.1.2.1.2.2.1.2.4 = STRING: "eth2"
iso.3.6.1.2.1.2.2.1.2.5 = STRING: "eth3"
iso.3.6.1.2.1.2.2.1.3.1 = INTEGER: 24
iso.3.6.1.2.1.2.2.1.3.2 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.3 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.4 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.5 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.4.1 = INTEGER: 65536
iso.3.6.1.2.1.2.2.1.4.2 = INTEGER: 1500
```

Q1.2 Retrieving SNMP data (Section 9.9, Exercise 1)

0.5 Points

Show the output of the `snmpget` commands.

— snmpget.png

```
ty2069@server:~$ snmpget -v 2c -c secret router-int IF-MIB::ifInUcastPkts.4
IF-MIB::ifInUcastPkts.4 = Counter32: 6619
ty2069@server:~$ snmpget -v 2c -c secret router-int IF-MIB::ifPhysAddress.4
IF-MIB::ifPhysAddress.4 = STRING: 2:44:f2:61:8f:d4
ty2069@server:~$ snmpget -v 2c -c secret router-int IF-MIB::ifOutUcastPkts.4
IF-MIB::ifOutUcastPkts.4 = Counter32: 6479
ty2069@server:~$
ty2069@server:~$ snmpget -v 2c -c secret router-int IF-MIB::ifDescr.4
IF-MIB::ifDescr.4 = STRING: eth2
```

Explain what each of the values you retrieved using `snmpget` means:

`ifDescr`, `ifInUcastPkts`, `ifPhysAddress`, `ifOutUcastPkts`.

(You can quote directly from the contents of

`/usr/share/snmp/mibs/ietf/IF-MIB`.)

`ifDescr` explanation:

"A textual string containing information about the interface. This string should include the name of the manufacturer, the product name and the version of the interface hardware/software."

`ifInUcastPkts` explanation:

"The number of packets, delivered by this sub-layer to a higher (sub-)layer, which were not addressed to a multicast or broadcast address at this sub-layer."

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of `ifCounterDiscontinuityTime`."

`ifPhysAddress` explanation:

The interface's address at its protocol sub-layer. For example, for an 802.x interface, this object normally contains a MAC address. The interface's media-specific MIB

B

must define the bit and byte ordering and the format of the value of this object. For interfaces which do not have such

an address (e.g., a serial line), this object should contain an octet string of zero length.

ifOutUcastPkts explanation:

The total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

Show the relevant section of the `ifconfig` output on the router. Annotate your screenshot: circle the interface information you collected using SNMP.

▼ router-ifconfig.png

 Download

```

ty2069@router-int:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.17.2.12 netmask 255.240.0.0 broadcast 172.31.255.255
    inet6 fe80::ae:41ff:fe4c:152d prefixlen 64 scopeid 0x20<link>
    ether 02:ae:41:4c:15:2d txqueuelen 1000 (Ethernet)
    RX packets 42804 bytes 47357788 (47.3 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 37192 bytes 3054038 (3.0 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.10.1.1 netmask 255.255.255.0 broadcast 10.10.1.255
    inet6 fe80::51:7ff:fe59:6305 prefixlen 64 scopeid 0x20<link>
    ether 02:51:07:59:63:05 txqueuelen 1000 (Ethernet)
    RX packets 216 bytes 14147 (14.1 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 82 bytes 8813 (8.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.10.2.1 netmask 255.255.255.0 broadcast 10.10.2.255
    inet6 fe80::44:f2ff:fe61:8fd4 prefixlen 64 scopeid 0x20<link>
    ether 02:44:f2:61:8f:d4 txqueuelen 1000 (Ethernet)
    RX packets 6619 bytes 505954 (505.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6479 bytes 626063 (626.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.10.3.1 netmask 255.255.255.0 broadcast 10.10.3.255
    inet6 fe80::60:a0ff:fea6:2863 prefixlen 64 scopeid 0x20<link>
    ether 02:60:a0:a6:28:63 txqueuelen 1000 (Ethernet)
    RX packets 140 bytes 9079 (9.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 17 bytes 2237 (2.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 14 bytes 1202 (1.2 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 14 bytes 1202 (1.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Q2 Telnet and SSH (Section 9.10, Exercise 4)

2 Points

Q2.1 telnet

1 Point

In the packet capture of the `telnet` experiment, which of the following can you read from the captured packets (i.e. not encrypted)? Select all that apply:



tcpdump

The screenshot displays a Wireshark interface with a packet capture of a Telnet session. The top section, 'Packet List', shows a single packet (No. 1) of type 'Telnet' from source '10.10.1.100' to destination '10.10.1.100'. The 'Packet Details' pane on the right shows the structure of the Telnet packet, including the 'Telnet Header' and 'Telnet Data'. The 'Packet Bytes' pane at the bottom shows the raw data of the packet, with a red circle highlighting the 'Telnet Data' field. The status bar at the bottom indicates 'Packets: 121 Displayed: 121 (100%)'.

The screenshot displays the Wireshark network protocol analyzer interface. The main window shows a packet capture of a Telnet session. The packet list pane on the left shows a list of captured packets, with the selected packet (No. 1) being a Telnet session from 10.10.1.1 to 10.10.1.2. The packet details pane on the right shows the structure of the selected packet, including the Telnet session header and the Telnet Data field. The packet bytes pane at the bottom shows the raw data of the packet, which is a Telnet session. The packet list pane shows the packet details, including the Telnet session structure. The packet details pane shows the Telnet session structure, including the Telnet Data field. The packet bytes pane shows the raw data of the packet, which is a Telnet session.

▼ telnet3.png

Download

The image shows a Wireshark packet capture of a Telnet session. The packet list on the left shows packets 34 through 53. Packet 34 is the first packet, and packet 53 is the last packet. The packet details pane shows the following information:

- Frame 36: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0
- Ethernet II, Src: 08:00:27:00:00:00 (08:00:27:00:00:00), Dst: 08:00:27:00:00:00 (08:00:27:00:00:00)
- Internet Protocol Version 4, Src: 10.10.1.100, Dst: 10.10.1.100
- Transmission Control Protocol, Src Port: 23, Dst Port: 23, Seq: 142, Ack: 90, Len: 10
- Telnet
- Data: Password:

The packet bytes pane shows the raw data of the packet, which is the password "Password".

▼ telnet4.png

Download

The image shows a Wireshark packet capture of a Telnet session. The packet list on the left shows packets 34 through 53. Packet 34 is the first packet, and packet 53 is the last packet. The packet details pane shows the following information:

- Frame 36: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface 0
- Ethernet II, Src: 08:00:27:00:00:00 (08:00:27:00:00:00), Dst: 08:00:27:00:00:00 (08:00:27:00:00:00)
- Internet Protocol Version 4, Src: 10.10.1.100, Dst: 10.10.1.100
- Transmission Control Protocol, Src Port: 23, Dst Port: 23, Seq: 90, Ack: 152, Len: 1
- Telnet
- Data: 4

The packet bytes pane shows the raw data of the packet, which is the character "4".

▼ telnet5.png

Download

The image shows a Wireshark packet capture of a Telnet session. The packet list on the left shows packets 80 through 96. Packet 80 is the first packet, and packet 96 is the last packet. The packet details pane shows the following information:

- Frame 91: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
- Ethernet II, Src: 08:00:27:00:00:00 (08:00:27:00:00:00), Dst: 08:00:27:00:00:00 (08:00:27:00:00:00)
- Internet Protocol Version 4, Src: 10.10.1.100, Dst: 10.10.1.100
- Transmission Control Protocol, Src Port: 23, Dst Port: 23, Seq: 1809, Ack: 187, Len: 20
- Telnet
- Data: PVI Src: 4 21:49:32 EST 2020

The packet bytes pane shows the raw data of the packet, which is the string "PVI Src: 4 21:49:32 EST 2020".

Q2.2 SSH

1 Point

In the packet capture of the SSH experiment, which of the following can you read from the captured packets (i.e. not encrypted)? Select all that apply:

- ☐ username and password
- ☒ IP headers (e.g. source and destination IP address)
- ☒ TCP headers (e.g. source and destination port)
- ☐ session data (e.g. commands that the user runs, like `date`, and command output)

To support your answer, upload screenshots from Wireshark or

`tcpdump`.

Annotate your screenshots to show *each* of the items above, and indicate whether you can read them (they are unencrypted) or not (they are encrypted).

▼ ssh.png [Download](#)

security-ssh-remote.pcap

File Edit View Go Capture Analysis Statistics Telephony Wireless Tools Help

Hosts & Network Filter: *SSH*

No.	Time	Source	Destination	Protocol	Length	Sequence number (bits)	Info
1	0.000000	10.10.1.100	10.10.2.100	TCP	74	44514 → 10000	[SYN] Seq=0 Win=0 Len=0
2	0.000000	10.10.1.100	10.10.2.100	TCP	74	10000 → 44514	[ACK] Seq=0 Win=0 Len=0
3	0.000000	10.10.1.100	10.10.2.100	TCP	60	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
4	0.000000	10.10.1.100	10.10.2.100	SSH	107	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0
5	0.000000	10.10.1.100	10.10.2.100	TCP	60	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0
6	0.000000	10.10.1.100	10.10.2.100	SSH	107	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
7	0.000000	10.10.1.100	10.10.2.100	TCP	60	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
8	0.000000	10.10.1.100	10.10.2.100	SSH	107	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0
9	0.000000	10.10.1.100	10.10.2.100	TCP	60	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
10	0.000000	10.10.1.100	10.10.2.100	SSH	107	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0
11	0.000000	10.10.1.100	10.10.2.100	TCP	60	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
12	0.000000	10.10.1.100	10.10.2.100	SSH	107	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0
13	0.000000	10.10.1.100	10.10.2.100	TCP	60	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
14	0.000000	10.10.1.100	10.10.2.100	SSH	107	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0
15	0.000000	10.10.1.100	10.10.2.100	TCP	60	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
16	0.000000	10.10.1.100	10.10.2.100	SSH	107	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0
17	0.000000	10.10.1.100	10.10.2.100	TCP	60	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
18	0.000000	10.10.1.100	10.10.2.100	SSH	107	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0
19	0.000000	10.10.1.100	10.10.2.100	TCP	60	44514 → 10000	[ACK] Seq=1 Ack=0 Len=0
20	0.000000	10.10.1.100	10.10.2.100	SSH	107	10000 → 44514	[ACK] Seq=1 Ack=0 Len=0

Frame 4: 107 bytes on wire (856 bits), 107 bytes captured (856 bits) on interface 0

Ethernet II, Src: 10.10.1.100, Dst: 10.10.2.100

Internet Protocol Version 4, Src: 10.10.1.100, Dst: 10.10.2.100

SSH (Secure Shell) Protocol

... .. Header Length: 20 bytes (5)

Total Length: 93

Identification: 0x00000000

Flags: 0x00000000, Don't Fragment

Fragment offset: 0

Time to live: 64

Protocol: TCP

Header checksum status: [Checksum disabled]

Source: 10.10.1.100

Destination: 10.10.2.100

Source Port: 44514

Destination Port: 10000

[SSH] Seq=1

[TCP] Seq=1

Sequence number: 1 (relative sequence number)

Sequence number (raw): 271700000

[Next sequence number: 2]

Acknowledgment number: 1 (relative ack number)

Acknowledgment number (raw): 433370000

1000 Header Length: 32 bytes (8)

Flags: 0x00000000, Don't Fragment

Window size value: 582

0000 02 51 07 50 03 00 02 f1 f4 52 1a 43 00 00 45 00 Q Y C R E

0001 00 5d 10 01 00 00 00 00 00 00 00 00 00 00 00 J 0 0 d

0002 02 04 ad e2 03 e8 a1 f9 01 8c 90 e7 74 d3 00 10 d 0 0 0 gt

security-ssh-remote.pcap

Packets: 76 / Displayed: 76 (100.0%)

Profile: Default

Q3 FTP and SFTP (Section 9.10, Exercise 6)

2 Points

Q3.1 FTP

1 Point

In the packet capture of the FTP experiment, which of the following can you read from the captured packets (i.e. not encrypted)? Select all that apply:

- ☒ username and password
- ☒ IP headers (e.g. source and destination IP address)
- ☒ TCP headers (e.g. source and destination port)
- ☒ session data (e.g. the name of the file that the user retrieves, the file contents)

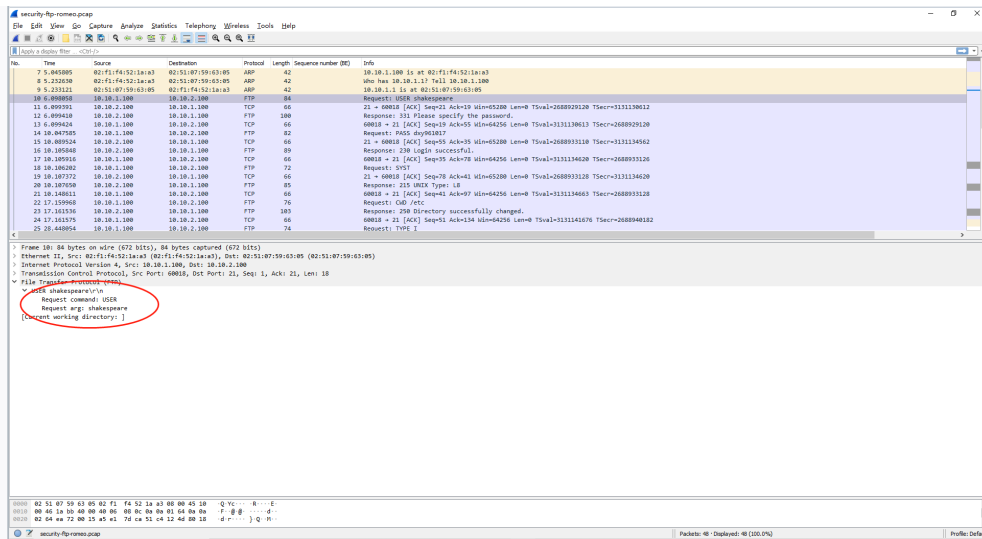
To support your answer, upload screenshots from Wireshark or `tcpdump`. Annotate your screenshots to show *each* of the items above, and indicate whether you can read them (they are unencrypted) or not (they are encrypted).

▼ ftp1.png [Download](#)

The screenshot shows a Wireshark packet capture of an FTP session. The packet list pane at the top shows 21 packets. Packet 21 is selected, and its details are shown in the packet details pane. The details pane shows the 'Request: USER Shakespeare' message. The packet bytes pane shows the raw data of the packet. The packet list pane shows the session flow from 1 to 21.

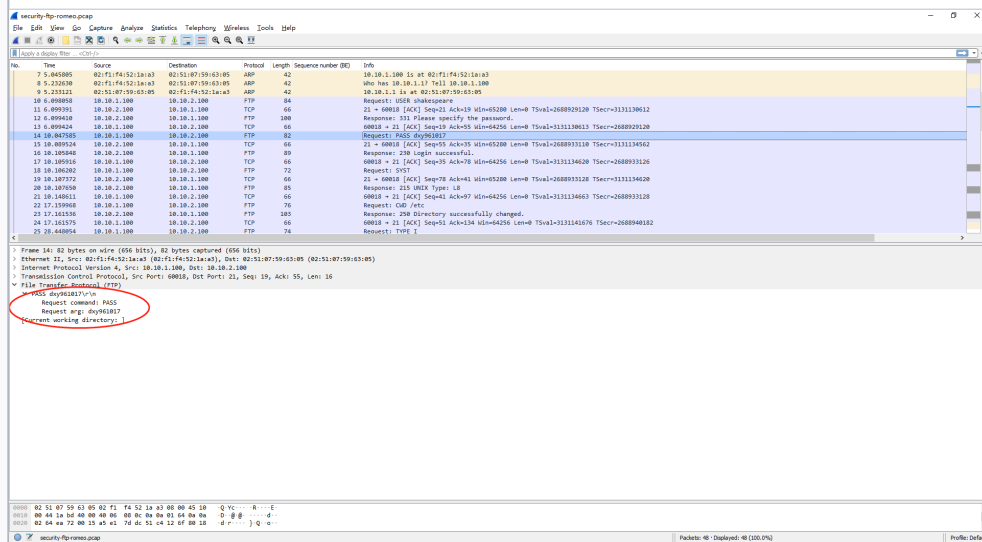
▼ ftp2.png [Download](#)

The screenshot shows a Wireshark packet capture of an FTP session. The packet list pane at the top shows 21 packets. Packet 21 is selected, and its details are shown in the packet details pane. The details pane shows the 'Request: USER Shakespeare' message. The packet bytes pane shows the raw data of the packet. The packet list pane shows the session flow from 1 to 21.



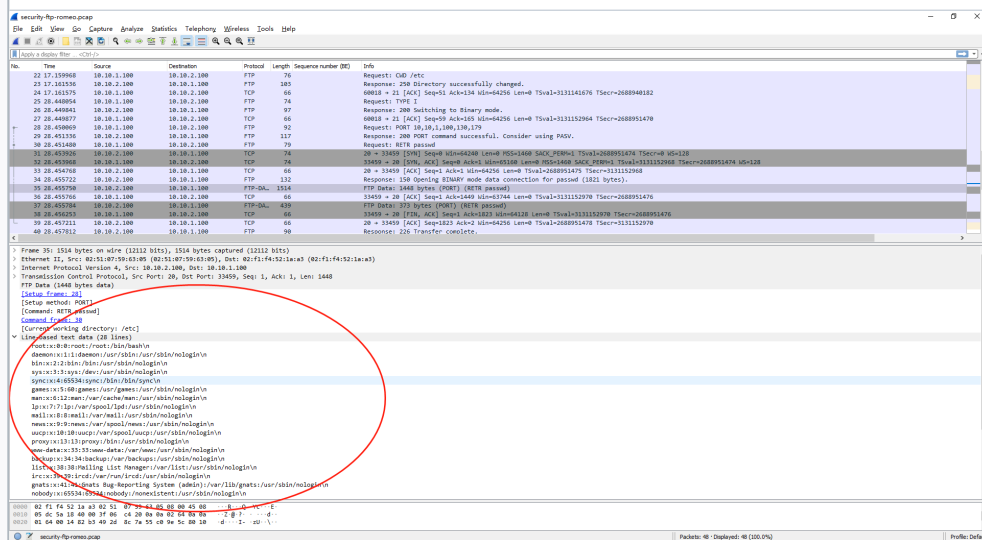
▼ ftp3.png

Download



▼ ftp4.png

Download



Q3.2 SFTP

1 Point

In the packet capture of the SFTP experiment, which of the following can you read from the captured packets (i.e. not encrypted)? Select all that apply:

☐ username and password

☒ IP headers (e.g. source and destination IP address)

☒ TCP headers (e.g. source and destination port)

☐ session data (e.g. the name of the file that the user retrieves, the file contents)

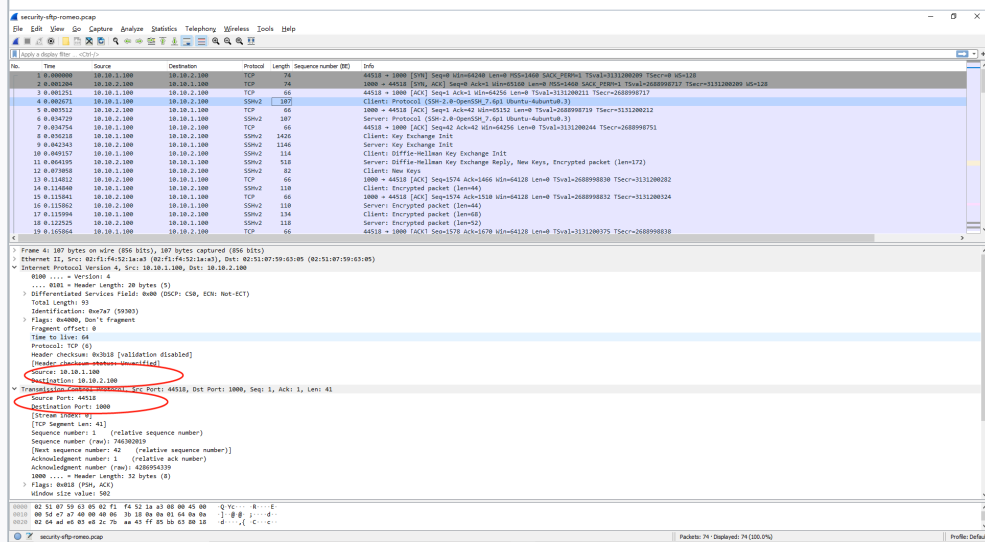
To support your answer, upload screenshots from Wireshark or

tcpdump.

Annotate your screenshots to show *each* of the items above, and indicate whether you can read them (they are unencrypted) or not (they are encrypted).

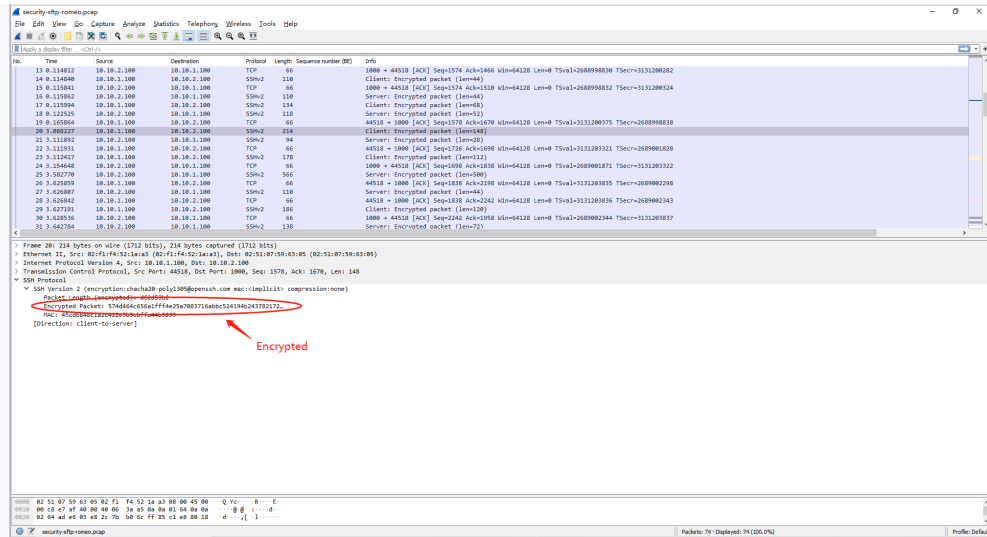
▼ sftp1.png

Download



▼ sftp2.png

Download



Q4 HTTP and HTTPS

2 Points

Q4.1 HTTP

1 Point

In the packet capture of the HTTP experiment, which of the following can you read from the captured packets (i.e. not encrypted)? Select all that apply:

- ☒ IP headers (e.g. source and destination IP address)
- ☒ TCP headers (e.g. source and destination port)
- ☒ contents of the HTTP GET (e.g. the name of the page you visited, `form.html`)
- ☒ session data (e.g. the data you entered into the form)

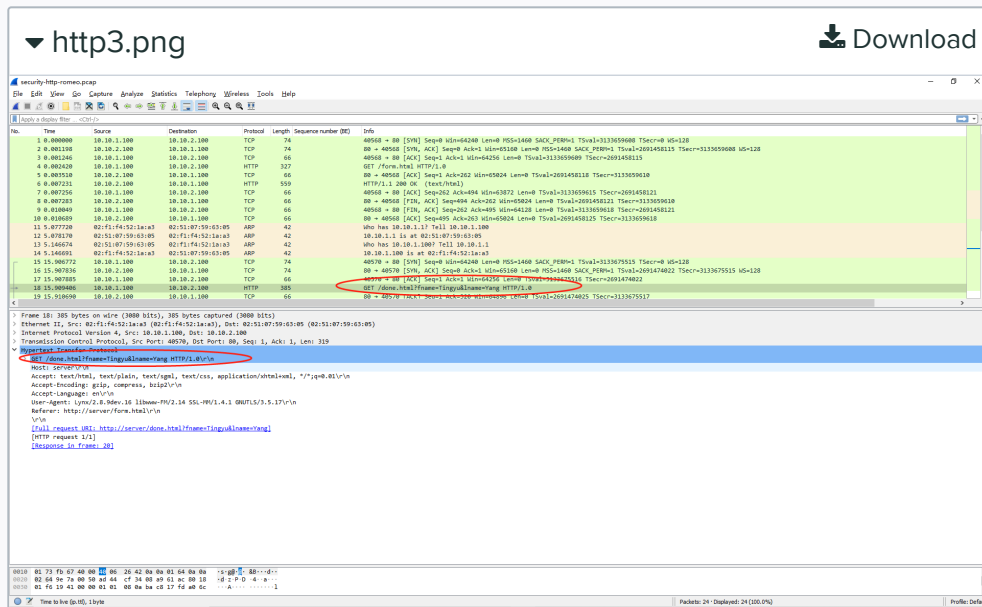
To support your answer, upload screenshots from Wireshark or

`tcpdump`.

Annotate your screenshots to show *each* of the items above, and indicate whether you can read them (they are unencrypted) or not (they are encrypted).

▼ http1.png

Download



Q4.2 HTTPS

1 Point

In the packet capture of the HTTPS experiment, which of the following can you read from the captured packets (i.e. not encrypted)? Select all that apply:

☒ IP headers (e.g. source and destination IP address)

☒ TCP headers (e.g. source and destination port)

☐ contents of the HTTP GET (e.g. the name of the page you visited, `form.html`)

☐ session data (e.g. the data you entered into the form)

To support your answer, upload screenshots from Wireshark or

`tcpdump`.

Annotate your screenshots to show *each* of the items above, and indicate whether you can read them (they are unencrypted) or not (they are encrypted).

▼ **https1.png** [Download](#)

The screenshot shows a Wireshark packet capture of an HTTPS connection. The packet list pane displays a list of packets, with the selected packet being a TCP Reset (RST) from 10.10.1.100 to 10.10.1.100. The packet details pane shows the TCP header with the 'RST' flag set and 'Sequence number: 1'. The packet bytes pane shows the raw data of the reset packet.

▼ **https2.png** [Download](#)

☒ IP address of "juliet" on the external network (not the address on the VPN tunnel!)

☒ IP address of the VPN server, "vpn" (i.e. the fact that the client is using this particular VPN server)

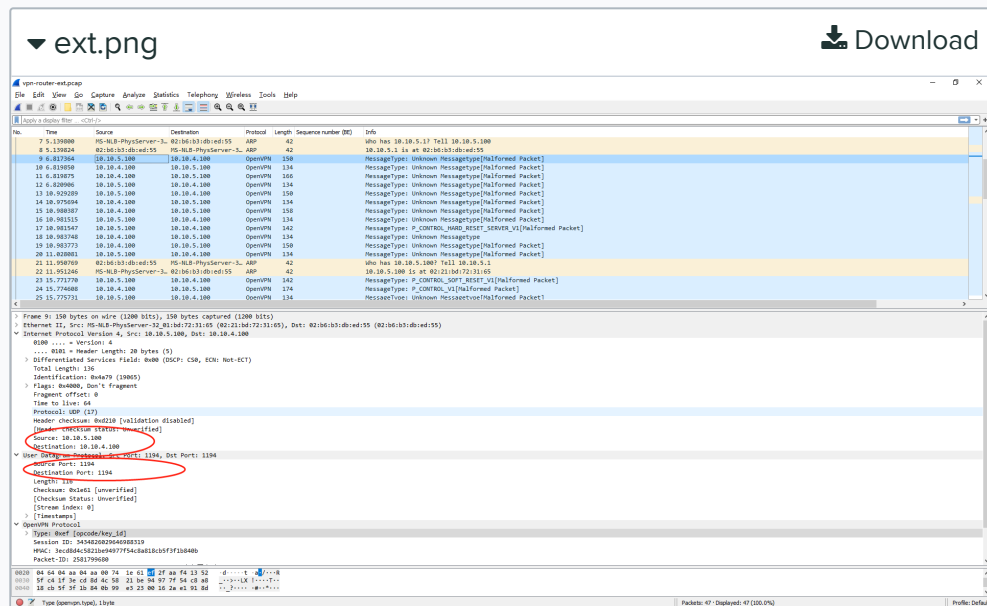
☐ IP address of the "server" node (i.e. the fact that the client is connecting to this particular FTP server)

☒ UDP port 1194 (i.e. this connection uses the well-known port number of OpenVPN, so eavesdroppers can identify it as VPN traffic)

☐ TCP port 21 (i.e. this connection uses the well-known port number of FTP, so eavesdroppers can identify it as FTP traffic)

☐ Session data (e.g. the name of the file that the user retrieves, the file contents)

To support your answer, upload screenshots from Wireshark or `tcpdump`. Annotate your screenshots to show *each* of the items above, and indicate whether you can read them (they are unencrypted) or not (they are encrypted).



Q5.2 VPN traffic at internal router

1 Point

In the packet capture on the *external router*, which of the following can you read from the captured packets?

This information would be visible to a potential eavesdropper located somewhere along the network path **between the VPN server and the FTP server**.

Select all that apply:

☐ IP address of "juliet" on the external network (not the address on the VPN tunnel!)

☒ IP address of the VPN server, "vpn" (i.e. the fact that the client is using this particular VPN server)

☒ IP address of the "server" node (i.e. the fact that the client is connecting to this particular FTP server)

☐ UDP port 1194 (i.e. this connection uses the well-known port number of OpenVPN, so eavesdroppers can identify it as VPN traffic)

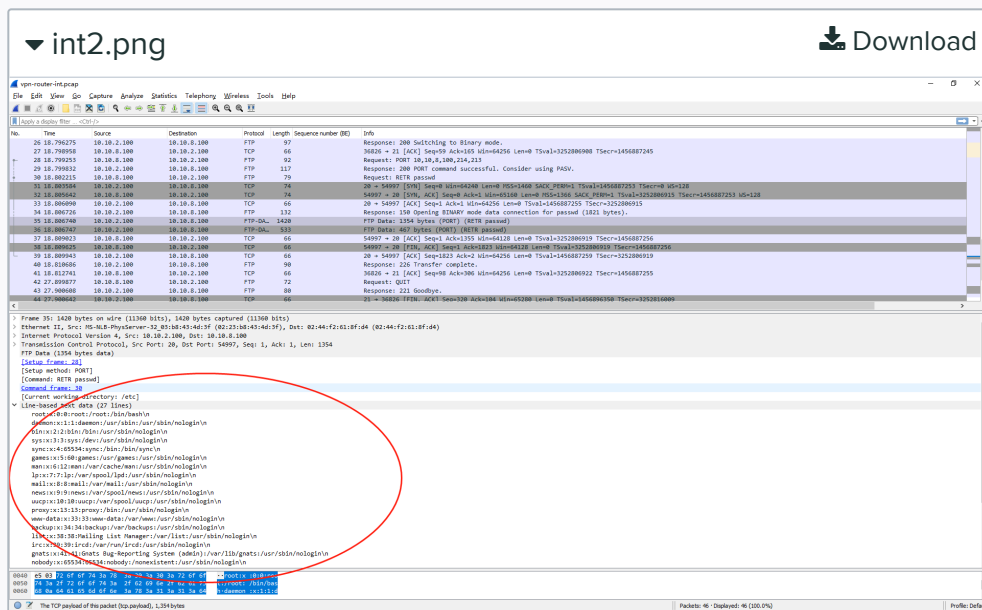
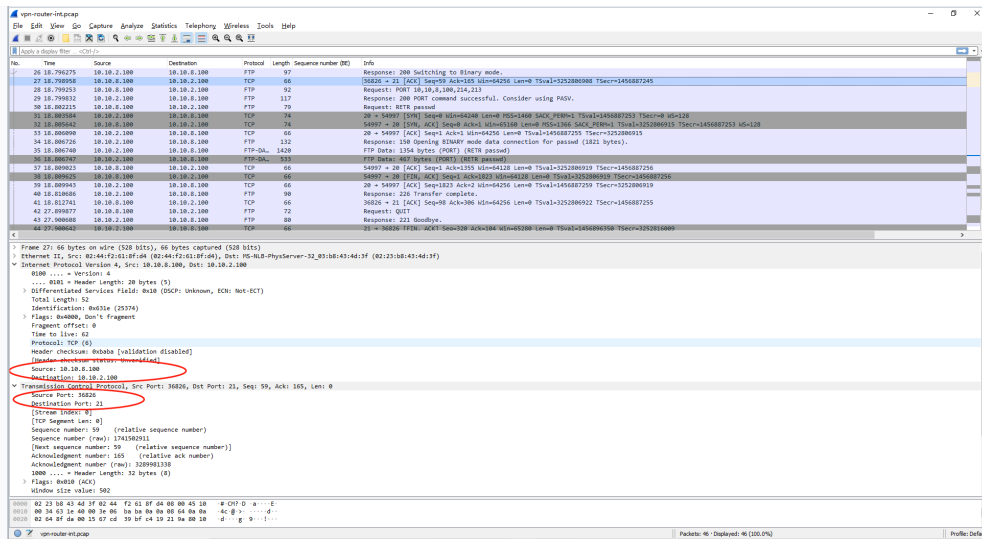
☒ TCP port 21 (i.e. this connection uses the well-known port number of FTP, so eavesdroppers can identify it as FTP traffic)

☒ Session data (e.g. the name of the file that the user retrieves, the file contents)

To support your answer, upload screenshots from Wireshark or `tcpdump`. Annotate your screenshots to show *each* of the items above, and indicate whether you can read them (they are unencrypted) or not (they are encrypted).

▼ int.png

 Download



Q6 Firewalls (Section 9.12, Exercises 8 & 9)

1 Point

Q6.1 Firewall with DROP (Section 9.12, Exercise 8)

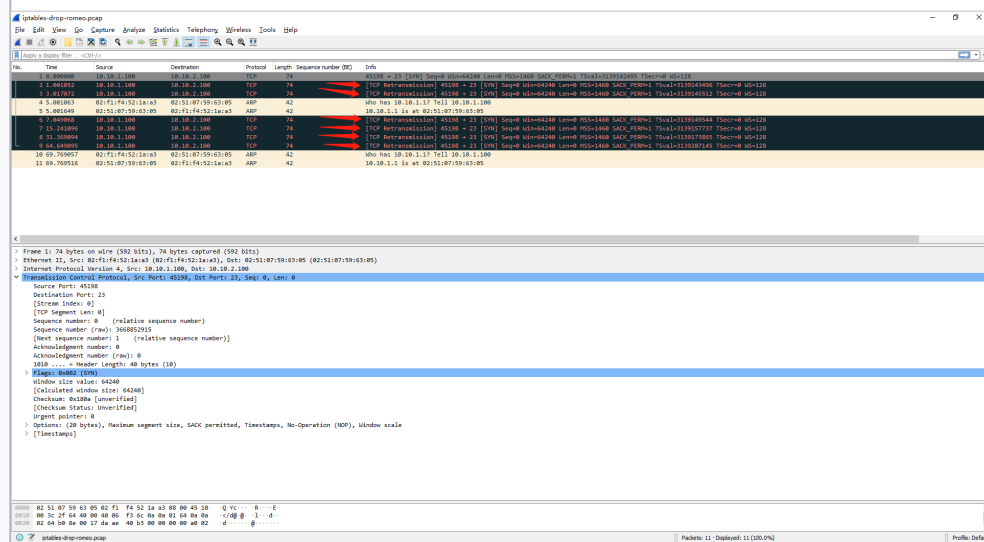
0.5 Points

Can you `telnet` to the "server" from "romeo"? Explain.

Use screenshots of your Wireshark or `tcpdump` output to show what happened. Annotate your screenshot: show where romeo attempts to initiate a `telnet` connection, and show what the response of the server is.

▼ drop.png

Download



We can't telnet to the server from romeo since in server's we added a rule to the end of the INPUT chain that drop all packet coming to server. So the packet from romeo never get to server, after a few retransmission, romeo stops the connection and output connection timed out.

Q6.2 Firewall with RST (Section 9.12, Exercise 9)

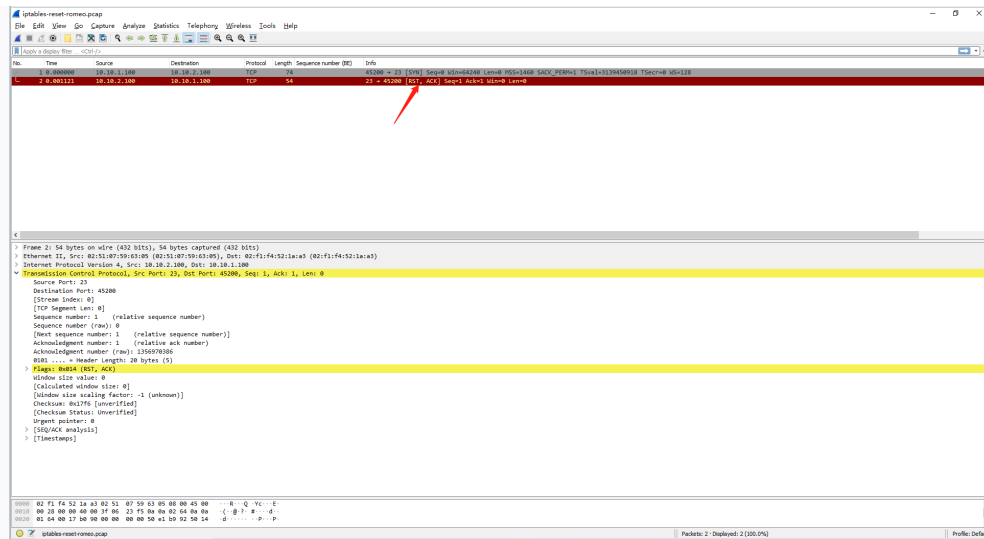
0.5 Points

Can you `telnet` to the "server" from "romeo"? Explain the difference between the `tcpdump` output in this exercise and the previous exercise.

Use screenshots of your Wireshark or `tcpdump` output to show what happened. Annotate your screenshot: show where romeo attempts to initiate a `telnet` connection, and show what the response of the server is.

▼ reset.png

Download



In this case, instead of drop the packet. server will send back a RST ACK and the connection will be refused.

Lab 9: Network management and Security

● **UNGRADED**

STUDENT

Tingyu Yang

TOTAL POINTS

- / 10 pts

QUESTION 1

SNMP (Section 9.9)

1 pt

1.1 — SNMP community string (Section 9.9, Exercise 2)

0.5 pts

1.2 — Retrieving SNMP data (Section 9.9, Exercise 1)

0.5 pts

QUESTION 2

Telnet and SSH (Section 9.10, Exercise 4)

2 pts

2.1 — telnet

1 pt

2.2 — SSH

1 pt

QUESTION 3

FTP and SFTP (Section 9.10, Exercise 6)

2 pts

3.1 — FTP

1 pt

3.2 — SFTP

1 pt

QUESTION 4

HTTP and HTTPS

2 pts

4.1 — HTTP

1 pt

4.2 — HTTPS

1 pt

QUESTION 5

Network layer security

2 pts

5.1 — VPN traffic at external router

1 pt

5.2 — VPN traffic at internal router

1 pt

QUESTION 6

Firewalls (Section 9.12, Exercises 8 & 9)

1 pt

6.1 — Firewall with DROP (Section 9.12, Exercise 8)

0.5 pts

6.2 — Firewall with RST (Section 9.12, Exercise 9)

0.5 pts