ANSWERS

# A12.4 The TCP/IP stack and protocols

Networks and the web · A level · OCR H446 1.3.3, AQA 7517 4.9.4.1, Eduqas A500QS 2.2 · about 30 min

BugBotLab

## What this lesson is about

The four layers, encapsulation, ports and sockets, and HTTP, HTTPS, FTP, SMTP, POP3 and SSH.

- Four layers
- Ports and sockets
- The application protocols
- A packet of your own

## Questions

**6 marks in all**

**1.** Put the layers of the TCP/IP stack in order, from the top (nearest the application) to the bottom. [1 mark]

Number the lines 1 to 4 to put them in the right order.

___ `Link`
___ `Network`
___ `Transport`
___ `Application`

**Answer:**

```
Application
Transport
Network
Link
```

Data passes down from the application layer through transport and network to the link layer, each adding its header.

**2.** Which layer adds the source and destination port numbers? [1 mark]

- ☐ **A** Transport
- ☐ **B** Application
- ☐ **C** Network
- ☐ **D** Link

**Answer:** A. The transport layer (TCP or UDP) uses ports to identify which process on each host the data belongs to.

**3.** What is a socket? [1 mark]

- ☐ **A** The combination of an IP address and a port number, one endpoint of a connection
- ☐ **B** A physical connector on a network card
- ☐ **C** A 48-bit hardware address
- ☐ **D** The list of rules in a firewall

**Answer:** A. For example 192.168.4.1:80; a connection is identified by the sockets at its two ends.

4. On which port does a web server listen for HTTPS by default? [1 mark]

> **Answer:** 443. HTTP uses port 80 and HTTPS uses 443.

5. Which protocol does an email client use to download messages from its mail server, usually deleting them from the server? [1 mark]

- ☐ **A** POP3
- ☐ **B** SMTP
- ☐ **C** FTP
- ☐ **D** SSH

> **Answer:** A. SMTP sends mail; POP3 retrieves it.

6. As a packet crosses several routers on the Internet, which addresses are replaced at every hop? [1 mark]

- ☐ **A** The MAC addresses in the link layer header
- ☐ **B** The IP addresses in the network layer header
- ☐ **C** The port numbers in the transport header
- ☐ **D** None of them change

> **Answer:** A. MAC addresses only have meaning on one local network, so each hop builds a new link layer header; the IP addresses stay the same (without NAT).

## The task: wrap it and unwrap it

The robot at `192.168.4.23` asks the Server at `192.168.4.1` for its battery level. Write `checksum(text)`: its input is any string; it returns the sum of the character codes (`ord`) of every character in `text`, modulo 256, as two uppercase hexadecimal digits (`format(n, "02X")`). 1. Build and `send` one packet: the body `<source IP>,<destination IP>,<source port>,<destination port>,<data>` with source port `49152`, destination port `80` and data `GET /battery`, then `*`, then the body's checksum. 2. The Server replies with a packet in the same format. Wait for it, checking `messages()` every 0.1 s. 3. Split off the checksum after the last `*`. If the body's checksum matches, print `checksum ok`, then `from socket <source IP>:<source port>`, then `data: <data>`. The data is everything after the fourth comma. If it does not match, print `checksum wrong`. The robot does not drive.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

def checksum(text):
    return "00"

send("192.168.4.23,192.168.4.1,49152,80,GET /battery")
```

**The hint students can ask for:** Write one function that works out a checksum for any text and use it both ways: to build your packet, and to check the reply. To unwrap the reply, split off the checksum at the star first, then split what is left at the commas, remembering the data is the last field.

### A solution

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

def checksum(text):
    total = 0
    for ch in text:
        total = total + ord(ch)
    return format(total % 256, "02X")

def packet(src_ip, dst_ip, src_port, dst_port, data):
    body = f"{src_ip},{dst_ip},{src_port},{dst_port},{data}"
    return body + "*" + checksum(body)

send(packet("192.168.4.23", "192.168.4.1", 49152, 80, "GET /battery"))
reply = None
while reply is None:
    wait(0.1)
    for sender, text in messages():
        reply = text
body, check = reply.rsplit("*", 1)
if checksum(body) == check:
    print("checksum ok")
    src_ip, dst_ip, src_port, dst_port, data = body.split(",", 4)
    print(f"from socket {src_ip}:{src_port}")
    print("data:", data)
else:
    print("checksum wrong")
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.