

Homework A

Solution Key

Problem A.1

Suppose that you wanted to send an email whose entire body was the following message:

`Très bien. => Ne donnez la traduction à personne !`

What character set would be appropriate to use? What would be the Content-Type (according to the MIME standard)? Assuming the message will be transmitted over plain SMTP (ruling out 8-bit transfer encodings), what would be the preferred Content-Transfer-Encoding for the message?

Using your chosen settings, give the byte-for-byte encoding of the message; what you hand in should be clear but also concise.

Any character set that can handle French would do, but the main candidates would be `iso-latin-1` (which also goes by the equivalent names `iso-8859-1` and `latin1`) or `UTF-8`. (The former is a strictly 8-bit set that handles the Western European languages and a few others; the latter is an 8-bit encoding of Unicode, which can theoretically handle any language at all.) The rest of the solution assumes `iso-latin-1`, but see below for some comments on Unicode.

The Content-Type line would then read

```
Content-Type: text/plain; charset=iso-latin-1
```

The two choices for Content-Transfer-Encoding are `quoted-printable` and `base64`, of which `quoted-printable` is preferred: to a mail reader that doesn't decode the message, it will still be mostly legible, and it only increases the message length by six bytes. (A `base64` encoding would be completely opaque, and would increase the message length by about 37%.)

The encoding of the message as `iso-latin-1` in `quoted-printable` is given below by printing the ASCII equivalent of each sent byte:

```
Tr=E8s bien. =3D) Ne donnez la traduction =E0 personne !
```

The only two characters in the original that are not ASCII are è and à, which are at code point 232 and 224 respectively in `iso-latin-1`. The `quoted-printable` encoding converts these to hexadecimal (0xe8 and 0xe0) and precedes them with an equals sign. The equals sign itself must also be encoded, and it is at code point 61 or 0x3d.

The Unicode solution is a little more complicated. Unicode itself is a multi-byte character set, with the super-duper-complete set requiring more than two bytes (as there are 17 ‘planes’ of 2^{16} character positions each). Unicode is usually transmitted encoded, either in UTF-16 (where each character is either two or four bytes) or UTF-8 (where each character is one to four bytes); the latter is typical for Latin-alphabet languages where most characters are in ASCII, as those characters transmit as a single byte. However, any non-ASCII character (such as è or à) gets encoded as at least two bytes, and in a message encoded with quoted-printable both of those bytes would need to be escaped. The quoted-printable UTF-8 version of the message would be as follows:

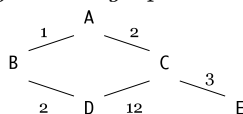
```
Tr=C3=A8s bien. =3D) Ne donnez la traduction =C3=A0 personne !
```

In practice, a Unicode-generating email client would probably generate this. However, there is yet another Unicode-based solution: the UTF-7 encoding was specifically designed for situations where Unicode needs to be transmitted over connections that aren’t 8-bit-clean. It uses a few tricks that basically let it base64-encode only those bytes that have to be encoded. If the charset is UTF-7, then no Content-Transfer-Encoding is needed; the message would be as follows:

```
Tr+A0g-s bien. =) Ne donnez la traduction +A0A personne !
```

Problem A.2

Consider the following subnet graph:



What is the best (shortest) route for a packet sent from node *C* to node *D*? Assume that the edge weights are stable and give the distance vector for node *C*...

The best route is C-A-B-D, because it is collectively of weight 5, while the “direct” route C-D is weight 12. The complete routing table for C (after the algorithm has converged) is

Dest	Dist	Via
A	2	A
B	3	A
C	0	–
D	5	A
E	3	E

Then, describe how the distance-vector routing algorithm would respond to the A-C link going down. In the immediately subsequent update, what would C’s new distance vector look like? How long would it take for all the routers in the system to converge on the new best routes?

To see what’s going on here, you need to first look at E’s original routing table:

Dest	Dist	Via
A	5	C
B	6	C
C	3	C
D	8	C
E	0	–

At the next update, C will receive E’s distance vector and assume that anything on E’s list is reachable from C via E with an additional cost of 3 (the link from C to E); that means that C’s next routing table will be the not-very-helpful

Dest	Dist	Via
A	8	E
B	9	E
C	0	–
D	11	E
E	3	E

After C's distance vectors have made another round trip through E, the routes via D start to look more attractive, but the spurious route to A remains:

Dest	Dist	Via
A	14	E
B	14	D
C	0	–
D	12	D
E	3	E

The routing table for E will then be almost correct on the next iteration, in that it would yield the correct route to everything (via C as before), but its estimated distance to A would be incorrect—17 instead of 18. Correcting this will require another full round trip, for a total of six updates since the link went down.

The situation on the other end of the broken link is even worse, because A and B will play the same game but will only ratchet up their distances by 2 per round trip, so it will require six round trips or twelve updates from the initial link failure before their distance vectors converge and always route to C and E via D.

Problem A.3

*Could a NAT box exist inside a network already behind a NAT?
How would that work, or why not?*

Yes (although there wouldn't be much point). Since the operation of a NAT box is essentially transparent to the computers it is connecting, a computer inside the inner NAT would have its IP address and port number translated into the IP address of the inner NAT and some arbitrary port number; and this IP address and port would themselves be translated to the address of the outer box and yet another arbitrary port.

However, given that the outer NAT box converts one IP address and 64K ports into 2^{24} IP addresses and 64K ports, it seems likely that the limiting factor will be the ports, not the IP addresses, so adding a nested NAT box is unlikely to help much.

Can a single network have multiple NAT gateways to the outside world? How, or why not?

Yes. When a host inside the NATted region sends out a packet, it might travel via either NAT box—which to that host appear to just be regular routers, and IP lets traffic flow over multiple valid routes. Once it goes through the NAT box, the source address/port translation occurs, and any response will pass via that NAT box and return to the original host. Subsequent outbound packets might travel via the other NAT box unless the network is carefully configured to prevent this, but while that would decrease the number of available ports, it would still *work*. (As I understand it, Knox's network uses some variant of this, which is why the whole campus appears to the outside world to comprise only a handful of IP addresses, as seen e.g. when posting on Wikipedia or the Wiki Fire.)