1. Un usuario navega desde una red corporativa, donde existe un DNS local recursivo. Sus tablas de traducción, junto con la de otros servidores DNS, se muestran abajo. Considera que el establecimiento de una conexión TCP consume un tiempo t_c, mientras que cualquier otro intercambio de información necesita t_m. [3]

   En esta situación, *a)* ¿qué tiempo sería necesario para que el usuario pudiera cargar en su navegador las siguientes URLs?. Ten en cuenta que las peticiones se realizan una tras otra, de forma secuencial, y que el tiempo de consulta al *resolver* de la máquina del usuario es despreciable. *b)* ¿Cómo queda la tabla de traducción del DNS local tras todas las resoluciones?.

   1. La página principal de YouTube.
   2. La página *http://www.mismascotas.com/carrito.php*. Considera que dicha página tiene 8 objetos embebidos, que el servidor soporta HTTP/1.1 y acepta únicamente 3 conexiones simultáneas del mismo cliente.
   3. La página *http://mail.mismascotas.com*.
   4. La página *http://www.cruzroja.org*.

DNS local

Dominio	Dirección IP
youtube.com	X.X.X.X
TLD .com	X.X.X.X
...	...

Servidor raíz E

Dominio	Dirección IP
TLD .com	X.X.X.X
TLD .org	X.X.X.X
TLD .net	X.X.X.X
...	...

TLD .com

Dominio	Dirección IP
nameserver.mismascotas.com	X.X.X.X
...	...

TLD .org

Dominio	Dirección IP
NS cruzroja.org	X.X.X.X
...	...

nameserver.mismascotas.com

Dominio	Dirección IP
mismascotas.com	X.X.X.X
mail.mismascotas.com	X.X.X.X
...	...

nameserver.cruzroja.org

Dominio	Dirección IP
cruzroja.org	X.X.X.X
...	...

Solución:

a) Para resolver el ejercicio, hay que conocer perfectamente la estructura del sistema DNS, e ir actualizando las tablas de resolución tras cada paso.

1. (**YouTube**) La resolución de YouTube se encuentra ya en el DNS local, por lo que el tiempo es simplemente la suma de t_m para la resolución, t_c para la conexión al servidor, y otro t_m para la descarga de la página índice, por lo que:

$$\boxed{t_1 = 2t_m + t_c}$$

   .
2. (**Carrito**) Ahora sí es necesario realizar la resolución completa, puesto que el dominio *mismascotas.com*, no se encuentra en el DNS local. La consulta a dicho DNS toma t_m. Éste tiene ya una entrada correspondiente al TDL de los dominios .com, por lo que directamente consulta a éste por el servidor primario del dominio (otro t_m), más una última consulta a este servidor primario (t_m). Por tanto, la resolución toma, $t_r = 3t_m$.
   Falta ahora calcular los tiempos de conexión y descarga de contenidos. Puesto que el servidor es HTTP/1.1, por defecto sus conexiones serán persistentes, aunque acepte únicamente 3 conexiones

simultáneas del mismo cliente. Una vez establecidas, los objetos se irán pidiendo alternativamente (en una especie de *round-robin*), para aprovechar al máximo todas las conexiones.
Finalmente, entra en juego el *pipelining*, que se usa por defecto en HTTP/1.1 si tanto el navegador como el servidor lo soportan. Resolvemos para ambos casos, que se aceptarían como válidos si se indica claramente qué opción se ha escogido.

Sin pipelining De esta forma, para descargar los objetos $O_1, ..., O_8$, los tiempos quedarían:

- Conexión 1 (t_{c1}): $t_c + O_1 + O_4 + O_7 = t_c + t_m + t_m + t_m = t_c + 3t_m$
- Conexión 2 (t_{c2}): $t_c + O_2 + O_5 + O_8 = t_c + t_m + t_m + t_m = t_c + 3t_m$
- Conexión 3 (t_{c3}): $t_c + O_3 + O_6 = t_c + t_m + t_m = t_c + 2t_m$

Estas conexiones y descargas ocurren simultáneamente, por lo que el tiempo de descarga de la página será:

$$t_d = \text{máx}\{t_{c1}, t_{c2}, t_{c3}\} = \text{máx}\{t_c + 3t_m, t_c + 3t_m, t_c + 2t_m\} = t_c + 3t_m$$

Finalmente, el tiempo total de descarga de la página 2 será:

$$\boxed{t_2 = t_r + t_d = 3t_m + t_c + 3t_m = t_c + 6t_m}$$

Con pipelining Ahora todos los objetos que se piden por la misma conexión se piden y cargan simultáneamente, por lo que tendríamos:

- Conexión 1 (t_{c1}): $t_c + O_1 + O_4 + O_7 = t_c + t_m$
- Conexión 2 (t_{c2}): $t_c + O_2 + O_5 + O_8 = t_c + t_m$
- Conexión 3 (t_{c3}): $t_c + O_3 + O_6 = t_c + t_m$

Ahora el tiempo de descarga de todos los objetos queda:

$$t_d = \text{máx}\{t_{c1}, t_{c2}, t_{c3}\} = \text{máx}\{t_c + t_m, t_c + t_m, t_c + t_m\} = t_c + t_m$$

Finalmente, el tiempo total de carga de la página 2 será:

$$\boxed{t_2 = t_r + t_d = 3t_m + t_c + t_m = t_c + 4t_m}$$

3. (**Mail**) Ahora se trata la resolución de un subdominio de *mismascotas.com*, cuyo servidor primario (*nameserver.mismascotas.com*) ya está en nuestro DNS local. Por tanto, éste consulta directamente al primario y obtiene la IP final:

$$\boxed{t_3 = \underbrace{t_m}_{\text{DNS local}} + \underbrace{t_m}_{\text{DNS primario}} + t_c + t_m = t_c + 3t_m}$$

4. (**Cruz Roja**) En este caso, el TLD del dominio .org no está en el DNS local, por lo que será necesario elevar la consulta hasta un servidor raíz. Por tanto, el tiempo de resolución será $t_r = 4t_m$. Resta solo añadir el tiempo de conexión y el de descarga de la página índice. Por tanto, queda $\boxed{t_4 = t_c + 5t_m}$.

b) Tras la resoluciones, la tabla del DNS local quedará:

DNS local

Dominio	Dirección IP
youtube.com	X.X.X.X
TLD .com	X.X.X.X
TLD .org	X.X.X.X
nameserver.mismascotas.com	X.X.X.X
mismascotas.com	X.X.X.X
mail.mismascotas.com	X.X.X.X
nameserver.cruzroja.org	X.X.X.X
cruzroja.org	X.X.X.X
...	...

2. Imagina una red P2P basada en el esquema de funcionamiento del protocolo Chord. Sin embargo, esta red utiliza unas funciones diferentes a las usuales para calcular los identificadores de nodo y contenido, que son: 3

   - $node\_id = (A + B + C + D) \mod 256$, donde A,..,D son los componentes de una dirección IP, A.B.C.D. Así, para la IP 138.100.10.100, A = 138, B = 100, C = 10 y D = 100.
   - $data\_id = (d_{n-3} + d_{n-2} + d_{n-1} + d_n) \mod 256$, donde d_i corresponde al código ASCII del carácter i-ésimo del contenido que quiere almacenarse, y n a su longitud.

   Para los siguientes nodos y contendios, ¿cómo quedaría la distribución de los mismos en la red? Calcula, asimismo, la tabla de *fingers* del nodo con IP '123.78.100.10' (en este caso, la función de cálculo sería mod 256). Detalla al máximo la resolución del ejercicio y realiza una representación gráfica del resultado. Las cadenas del contenido son sensibles a mayúsculas y minúsculas.

Nodos
132.44.100.10
66.30.58.50
201.49.10.100
77.144.23.32
123.78.100.10
123.78.100.11

Contenido
Amazon Alexa
Oracle Cloud
Idealista
UAM.es

Figura 1: Tabla ASCII

Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex
(sp)	32	0040	0x20	@	64	0100	0x40	`	96	0140	0x60
!	33	0041	0x21	A	65	0101	0x41	a	97	0141	0x61
"	34	0042	0x22	B	66	0102	0x42	b	98	0142	0x62
#	35	0043	0x23	C	67	0103	0x43	c	99	0143	0x63
$	36	0044	0x24	D	68	0104	0x44	d	100	0144	0x64
%	37	0045	0x25	E	69	0105	0x45	e	101	0145	0x65
&	38	0046	0x26	F	70	0106	0x46	f	102	0146	0x66
'	39	0047	0x27	G	71	0107	0x47	g	103	0147	0x67
(	40	0050	0x28	H	72	0110	0x48	h	104	0150	0x68
)	41	0051	0x29	I	73	0111	0x49	i	105	0151	0x69
*	42	0052	0x2a	J	74	0112	0x4a	j	106	0152	0x6a
+	43	0053	0x2b	K	75	0113	0x4b	k	107	0153	0x6b
,	44	0054	0x2c	L	76	0114	0x4c	l	108	0154	0x6c
-	45	0055	0x2d	M	77	0115	0x4d	m	109	0155	0x6d
.	46	0056	0x2e	N	78	0116	0x4e	n	110	0156	0x6e
/	47	0057	0x2f	O	79	0117	0x4f	o	111	0157	0x6f
0	48	0060	0x30	P	80	0120	0x50	p	112	0160	0x70
1	49	0061	0x31	Q	81	0121	0x51	q	113	0161	0x71
2	50	0062	0x32	R	82	0122	0x52	r	114	0162	0x72
3	51	0063	0x33	S	83	0123	0x53	s	115	0163	0x73
4	52	0064	0x34	T	84	0124	0x54	t	116	0164	0x74
5	53	0065	0x35	U	85	0125	0x55	u	117	0165	0x75
6	54	0066	0x36	V	86	0126	0x56	v	118	0166	0x76
7	55	0067	0x37	W	87	0127	0x57	w	119	0167	0x77
8	56	0070	0x38	X	88	0130	0x58	x	120	0170	0x78
9	57	0071	0x39	Y	89	0131	0x59	y	121	0171	0x79
:	58	0072	0x3a	Z	90	0132	0x5a	z	122	0172	0x7a
;	59	0073	0x3b	[	91	0133	0x5b	{	123	0173	0x7b
<	60	0074	0x3c	\	92	0134	0x5c	|	124	0174	0x7c
=	61	0075	0x3d	]	93	0135	0x5d	}	125	0175	0x7d
>	62	0076	0x3e	^	94	0136	0x5e	~	126	0176	0x7e
?	63	0077	0x3f	_	95	0137	0x5f				