CAPÍTULO 1
ESCENARIO

Luis Marrone

Como anticipamos en el prefacio les proponemos como escenario para derribar esos mitos y tabúes de TCP/IP un simple acceso a `www.unlp.edu.ar`. Suponemos que la conocen, ¿no?.

Para llevar adelante ese cometido tenemos que disponer de un navegador web como comúnmente se le llama que para ser precisos (vayámonos acostumbrando a esto), no es otra cosa que un programa cliente. Cliente porque su desarrollo está basado en el paradigma cliente-servidor. Más allá de sus diferencias en cuanto a velocidad, funcionalidad, servicios, seguridad, etc.., todos ellos, llámense Firefox, Chrome, Internet Explorer, Safari, Opera, etc.., se basan en este paradigma.

Este paradigma es con el que se desarrollaron las primeras aplicaciones o servicios a través de Internet. Luego aparecieron otros como "peer-to-peer", que en la actualidad compite en intensidad de tráfico con cliente-servidor.

Vayamos elaborando el reparto de todos los actores necesarios para completar nuestro sencillo acceso a la página web de la Universidad. Hasta ahora estamos:

Actores	
1	Programa-Servicio-Aplicación

El texto de este actor será *www.unlp.edu.ar* que incluye en el nombre de la página que queremos acceder la ubicación del servidor, el otro integrante del paradigma de programación de la aplicación. En el mundo de TCP/IP la ubicación de este actor está dada por lo que llamamos la dirección IP. Puede que el navegador no la conozca, es decir, no la tenga disponible en su memoria. En ese caso, llamará a un actor de reparto que le va a conseguir la dirección IP que no tiene. Este actor de reparto es la aplicación/servicio **DNS** [Moc87], [Moc89] y [THKS03]. Lo llamamos actor de reparto por cuanto es un servicio auxiliar. Nosotros, usuarios, difícilmente querremos acceder directamente a este servicio. Se incrementa el reparto:

Actores	
1	Programa-Servicio-Aplicación
2	DNS (Domain Name Service)

Ya sabemos la página que queremos acceder y su dirección IP. Tendremos ahora que enviarle un mensaje al servidor para pedirle el acceso y que nos envíe esa página. Por lo pronto construiremos un mensaje con ese pedido tal que lo comprenda el servidor. Para esto, ese mensaje tendrá un formato definido por un protocolo en particular. No existe un único protocolo. En nuestro ejemplo tomaremos **HTTP**, HyperText Transfer Protocol. [BLFF96]

Tenemos que conseguir que ese pedido llegue al servidor. El programa servidor reside en un host o dispositivo de hardware generalmente destinado a ejecutar ese programa y que estará recibiendo múltiples pedidos como el nuestro. No está de más invitarlo a que nos atienda. Vamos a avisarle que queremos solicitarle información que tiene disponible y que es de nuestro interés. Esperamos que acepte. Para poder cumplir con todo eso vamos a recurrir al envío de un mensaje definido por un protocolo **TCP**, (Transmission Control Protocol) [Pos81c], que contendrá al mensaje anterior de **HTTP**. Este protocolo llevará un control estricto del envío, detectando errores y generando retransmisiones en esos casos. En otras palabras, nos garantiza la recepción de lo que envía.

Ya que mencionamos varias veces a un protocolo es hora de definirlo, por lo menos protocolos que son de nuestro interés:

> **protocolo:** Conjunto de reglas y tipos de mensajes que permiten el intercambio de información entre dispositivos lejanos y con un nivel de confiablilidad y seguridad definido ad-hoc.

Nuestro reparto sigue creciendo:

	Actores
1	Programa-Servicio-Aplicación
2	DNS (Domain Name Service)
3	HTTP
4	TCP

En otras ocasiones, el mensaje que necesitamos transferir es el que corresponde a tráfico de voz o de video que pertenecen a una clase particular de tráfico que es el de tiempo real. Ese tráfico requiere un envío inmediato y no necesita de un control tal que frente a un pérdida lo retransmitamos. Una retransmisión, por ejemplo, generaría un ruido en nuestros oídos. Es más, por su naturaleza enviamos el mensaje directamente, no dialogamos con el servidor previamente como en los otros casos. Por eso, en ese tipo de tráfico de tiempo real, necesitamos acudir a otro protocolo con una funcionalidad que se ajusta a esas características. El protocolo es UDP (User Datagram Protocol), [Pos80]. Estas características hacen que sea utilizado, también, por el DNS, por lo que lo hace un actor importante para nuestro análisis.

	Actores
1	Programa-Servicio-Aplicación
2	DNS (Domain Name Service)
3	HTTP
4	TCP-UDP

Así como queremos conectarnos con el servidor de la Universidad, también, pretendemos tener una amplia versatilidad en las conexiones. Poder conectarnos con servidores tanto próximos como lejanos. En otras palabras, independizarnos de la distancia a la que nos encontremos del servidor. En ese caso vamos a recurrrir a un servicio dado por una red como Internet y, para ello, a nuestro mensaje TCP habremos de encapsularlo en otro mensaje que nos permita disponer de esos servicios de red. Lo encapsularemos en el protocolo **IP** [Pos81a] y [DH17], que tiene dos versiones vigentes, IPv4 e IPv6, presentes en la red de redes como Internet. Los actores siguen creciendo:

Actores	
1	Programa-Servicio-Aplicación
2	DNS (Domain Name Service)
3	HTTP
4	TCP-UDP
5	IP

Podemos resumir en una palabra los servicios de red que mencionamos: *"conectividad IP"*.

conectividad IP: Propiedad del protocolo IP que permite el intercambio de datagramas entre usuarios de redes IP interconectadas. Para lograrlo el protocolo IP necesita solo conocer las direcciones IP origen y destino de los usuarios

Esas direcciones IP que identifican a los usuarios de una red IP y que como recordarán, con el mismo formato identificamos a una red IP, son parámetros de configuración entre otros. Configuración que la hacemos manualmente o estáticamente como suele decirse en la jerga local de IPv4 o dinámicamente como en IPv6.

También podemos recurrir a un servicio como **DHCP** [Dro97] y [HFZT10] que nos entrega la dirección IP y toda la información que necesitamos para lograr esa conectividad.

Para finalizar con este breve pantallazo de IP vale la pena recordar que es un protocolo sin conexión, los datagramas, así se los llama a los mensajes de IP, son independientes, es como un servicio de correo postal simple. Para que el mensaje llegue lo incluyo en un sobre con la dirección destino e introduzco el sobre en el buzón más cercano. No sé si el mensaje llegó a destino. Con IP ocurre lo mismo. Por eso aquello de *best effort* que lo caracteriza.

Pero contamos con un protocolo auxiliar como **ICMP** [Pos81b] y [CD06] que como vamos a ver brinda algo de gestión a las redes IP.

¿Cómo accedemos a Internet? Tenemos que llegar de nuestro equipo de cómputo a "La Red". Independientemente de la ubicación del equipo, este formará parte de una red LAN que lo conectará al router de salida/ingreso a Internet de nuestro lugar de trabajo o a un port del modem-switch-router que nos instaló nuestro proveedor de Internet, ISP, en nuestro hogar. La LAN logra que el mensaje IP llegue al dispositivo de salida/acceso a Internet. El mensaje IP (datagrama) tendrá que encapsularse en un mensaje del protocolo de LAN, típicamente Ethernet [Bog76]. Hará falta conocer la dirección Ethernet del dispositivo de salida, que si no la conoce nuestro equipo acudirá a protocolos auxiliares como **ARP** [Plu82] para que obtenga esa dirección Ethernet. Completamos así el reparto:

Actores	
1	Programa-Servicio-Aplicación
2	DNS (Domain Name Service)
3	HTTP
4	TCP-UDP
5	IP-ICMP-DHCP
6	Ethernet-ARP

Este es el escenario propuesto con sus actores principales. Nuestras actividades nos llevan en muchas ocasiones a analizar el comportamiento/performance de cada uno de ellos así como su conjunto y, también, producto de ese análisis puede que necesitemos modificar su diseño para optimizar la performance.

Actividades estas que suelen llevarse a cabo de una mejor manera si contamos con un modelo de ese escenario.

En nuestro caso, como sabemos, el modelo más comúnmente empleado es el modelo TCP/IP. Recordamos también como se distribuyen esos actores en los niveles/capas del modelo:

- Modelo TCP/IP
  - Nivel de Aplicación
    - Programa-Servicio-Aplicación
    - DNS (Domain Name Service)
    - HTTP
  - Nivel de Transporte
    - TCP
    - UDP
  - Nivel de Red
    - IP
    - ICMP
    - DHCP
  - Nivel de acceso al medio
    - Ethernet
    - ARP

Aprovechamos para recordar que el nombre proviene del protocolo del nivel de red como IP y de uno de los protocolos del nivel de transporte como TCP. **TCP/IP no es un protocolo**. Para llevar a cabo un correcto análisis y no generar confusión alguna es importante que seamos precisos. Por ello esa aclaración.

REFERENCIAS

[BLFF96] T. Berners-Lee, R. Fielding y H. Frystyk. «Hypertext transfer protocol – HTTP/1.0», 1996.

[Bog76] Robert Metcalfe; David Boggs. «Ethernet: Distributed packet switching for local computer networks». *Communications of the ACM*, vol. 19 nº 7, páginas 395–405, 1976.

[CD06] A. Conta y S. Deering. «Internet control message protocol (ICMPv6) for the internet protocol version 6 (IPv6) specification», 2006.

[DH17] S. Deering y R. Hinden. «Internet protocol, version 6 (ipv6) specification», 2017.

[Dro97] R. Droms. «Rfc 2131: Dynamic host configuration protocol», 1997.

[HFZT10] T. Huth, J. Freimann, V. Zimmer y D. Thaler. «Rfc-5970: Dhcpv6 options for network boot», 2010.

[Moc87] P. V. Mockapetris. «Rfc 1035: Domain names: Implementation specification», 1987.

[Moc89] P. V. Mockapetris. «Rfc-1101: Dns encoding of network names and other types», 1989.

[Plu82] David C. Plummer. «Rfc 826: An ethernet address resolution protocol (arp)», 1982.

[Pos80] Jon Postel. «Rfc-768: User datagram protocol (udp)», 1980.

[Pos81a] Jon Postel. «Rfc-791: Internet protocol (ip)», 1981.

[Pos81b] Jon Postel. «Rfc-792: Internet control message protocol (icmp)», 1981.

[Pos81c] Jon Postel. «Rfc-793: Transmission control protocol (tcp)», 1981.

[THKS03] S. Thomson, C. Huitema, V. Ksinant y M. Souissi. «Rfc 3596: Dns extensions to support ip version 6», 2003.