Blandskron

Funcionamiento Interno de la Conexión Web Segura: De la Capa Aplicación a la Física

Cuando un usuario escribe una URL como `https://intranet.miempresa.local` en su navegador, se desencadena una secuencia estrictamente estructurada de procesos que involucran cada capa del modelo TCP/IP. Esta lectura descompone esa operación con precisión, mostrando cada actor técnico involucrado.

Resolución de Nombre (DNS) – Capa de Aplicación

Antes de cualquier conexión, es necesario conocer la dirección IP del servidor destino. El protocolo DNS (Domain Name System) realiza esta traducción. En un entorno real o simulado, el sistema operativo consulta su caché de DNS local o realiza una petición a un servidor configurado (como 8.8.8.8). Si la entrada no se encuentra, la consulta se resuelve recursivamente.

En entornos simulados como laboratorios locales, la resolución puede forzarse directamente mediante el archivo `hosts`, lo que evita la consulta al DNS externo. En ese archivo, se define la IP asociada al dominio local, por ejemplo:

127.0.0.1 intranet.miempresa.local

Esto garantiza que cualquier aplicación que intente resolver `intranet.miempresa.local` obtenga inmediatamente `127.0.0.1`.

Establecimiento de Conexión TCP – Capa de Transporte

Una vez resuelto el nombre de dominio, el navegador solicita al sistema operativo abrir una conexión TCP con la IP destino en un puerto específico, típicamente 443 para HTTPS. El protocolo TCP inicia un *three-way handshake*:

1. Cliente envía un paquete SYN (synchronize)
2. Servidor responde con SYN-ACK
3. Cliente responde con ACK

Esto establece una conexión de sesión orientada a conexión, confiable y de flujo controlado.

Durante esta fase se establecen los identificadores de sesión, número de secuencia inicial, tamaño de ventana TCP y otras opciones como SACK (Selective Acknowledgement) y escalado de ventana.

Negociación TLS – Capa de Aplicación (sobre TCP)

Al utilizar HTTPS, una vez completado el *handshake* TCP, se inicia inmediatamente la negociación TLS (Transport Layer Security). Este protocolo criptográfico asegura que toda la comunicación posterior esté cifrada.

Proceso resumido:

1. Cliente envía un *ClientHello*, especificando versiones TLS soportadas, suites de cifrado preferidas, extensiones como SNI (Server Name Indication) y un nonce aleatorio.

2. Servidor responde con *ServerHello*, seleccionando parámetros comunes, enviando su certificado X.509 con su clave pública, un nuevo nonce y firmando el contenido con su clave privada.

3. Cliente verifica el certificado, genera una clave simétrica, la cifra con la clave pública del servidor y la envía.

4. A partir de este punto, toda la comunicación se cifra con esa clave simétrica (generalmente usando AES en modo GCM).

Si el certificado no puede verificarse, el navegador bloquea la conexión, evitando ataques man-in-the-middle.

Solicitud HTTP – Capa de Aplicación (sobre TLS)

Con el canal seguro establecido, el navegador construye y envía una solicitud HTTP utilizando el método adecuado (GET, POST, etc.) con encabezados como `Host`, `User-Agent`, `Accept`, y cookies si corresponde.

La solicitud está encapsulada dentro del túnel TLS, que a su vez está encapsulado en TCP, asegurando confidencialidad e integridad.

Ejemplo de payload (cifrado en tránsito):

GET /home HTTP/1.1

Host: intranet.miempresa.local

Accept: text/html

El servidor responde con un código HTTP (ej. 200 OK) y el contenido, también cifrado bajo TLS.

Capa de Internet – IP y Encaminamiento

Cada segmento TCP es encapsulado en un datagrama IP. El encabezado IP incluye:

- Dirección IP de origen y destino
- TTL (Time To Live)
- Protocolo superior (TCP)
- Fragmentación si es necesario

El datagrama IP se enruta desde el cliente hasta el servidor pasando por gateways y routers intermedios, según su tabla de enrutamiento. En redes privadas (como 192.168.0.0/16), el router realiza NAT si es necesario.

Capa de Enlace y Física – Tramas Ethernet y Señal Eléctrica

Finalmente, el datagrama IP es encapsulado en una trama Ethernet para ser transmitido por la red local. Esta contiene:

- MAC de origen y destino
- Tipo de protocolo (IPv4 → 0x0800)
- Payload: el datagrama IP

En redes cableadas, la transmisión se realiza mediante señales eléctricas moduladas sobre el cable. En redes inalámbricas, la transmisión es mediante modulación de ondas de radio con protocolos como IEEE 802.11.

Cada host decide si aceptar la trama al verificar si su MAC coincide o si está en modo promiscuo (como en sniffing con Wireshark).

Respuesta del Servidor y Flujo Reverso

El servidor repite el proceso de forma inversa:

1. Construye la respuesta HTTP cifrada bajo TLS.
2. Encapsula en TCP/IP.
3. Envía el datagrama IP, luego encapsulado en Ethernet.
4. Llega al cliente, quien desencapsula capa por capa hasta entregar la respuesta al navegador.

Encapsulación Técnica: Detalle de Paquetes

Cada solicitud HTTPS viaja de forma encapsulada en múltiples niveles:

[Nivel físico] Señales eléctricas

↳ [Enlace] Trama Ethernet (MAC origen/destino)

↳ [Internet] Paquete IP (IP origen/destino)

↳ [Transporte] Segmento TCP (puertos, ACK, ventanas)

↳ [Aplicación] Registro TLS (cifrado)

↳ [Aplicación] Solicitud/respuesta HTTP

Cada capa solo ve su contexto. Por ejemplo:

- El router solo procesa el encabezado IP.
- El firewall de capa 7 inspecciona HTTP (si tiene capacidad de inspección profunda).
- El navegador solo ve la respuesta HTTP una vez que TLS la descifra.

Consideraciones de Seguridad

1. **Integridad:** Garantizada por hashes y MACs (HMAC-SHA256) en TLS.
2. **Confidencialidad:** Clave simétrica AES-256 evita que terceros vean el contenido.
3. **Autenticación del servidor:** Verificada mediante la cadena de confianza del certificado TLS.
4. **Mitigación de ataques:** TLS mitiga sniffing, MITM y modificación de paquetes. DNSSEC podría reforzar la autenticación del origen en DNS.

Implementación Local Controlada

Para entornos de laboratorio, como los ejercicios prácticos planteados, se puede simular completamente esta cadena:

- Entrada personalizada en `/etc/hosts`
- Contenedor Docker que sirve una app web básica
- Traefik como gateway que enruta peticiones
- Inspección de tráfico con DevTools o Wireshark

La conexión `http://intranet.miempresa.local` pasa por Traefik, que actúa como reverse proxy, y enruta la petición al contenedor correspondiente. Las rutas se definen mediante etiquetas (labels) en el `docker-compose.yml`, y pueden incluir middleware como `StripPrefix` para modificar la URL antes de redirigirla.