Configuración de hosts

Introducción

La configuración de hosts consiste en darle a un host que está booteando sus parámetros de red. Este host puede no saber su propia dirección IP o máscara, por lo que el procedimiento debe poder lograrse a pesar de no tener un stack completo. En ciertos casos, es hasta posible entregarle a un host la imagen de sistema operativo que debe ejecutar, permitiendole bootear a una estación completamente stateless.

RARP y BOOTP

El primer parámetro a obtener es la dirección IP. Antiguamente este era el único parámetro indispensable, ya que la máscara de red se podía deducir de la dirección. Hoy en día esto no es suficiente. Para obtener una dirección, se utilizaba RARP (ver RFC-903), que es una variante de ARP. RARP consiste en una petición ARP, indicando que dada una dirección de hardware se desea obtener una dirección IP. El host anunciaría su propia dirección ethernet y un servidor RARP le devolvería su dirección. Este protocolo tiene varias limitaciones. En primer lugar, solamente puede devolver una dirección IP, lo que era limitado en su momento y hoy es insuficiente. En segundo lugar, los pedidos ARP viajan directamente sobre el nivel de enlace, lo que hace necesario un servidor RARP en cada subred. Este protocolo también está limitado a siempre devolver la misma dirección a cada host (definido por su MAC address), ya que no tiene forma de desasignar una dirección entregada.

BOOTP (Bootstap Protocol, RFC-951) fue creado para cubrir las deficiencias de RARP. BOOTP trabaja sobre UDP (un protocolo de nivel de transporte, no orientado a conexión, orientado a datagramas y no confiable), lo que le permite (con ayuda de ciertos dispositivos) viajar más allá del segmento local. BOOTP también facilita la programación de los servidores, ya que puede aprovechar el stack TCP/IP del servidor si el daemon BOOTP puede agregar una entrada en la tabla de ARP. BOOTP trabaja sobre los puertos UDP/67 y UDP/68.

Un paquete BOOTP es esencialmente un formulario que contiene (entre otras cosas) lugar para la dirección del servidor, del cliente, un número de transacción y un nombre de archivo de bootup. En el caso más básico, el cliente envía el formulario a la dirección de broadcast local (255.255.255.255) y un servidor BOOTP local lo recibe, completa los parámetros faltantes y lo devuelve a la dirección de nivel 2 indicada. Una vez terminada esta transacción, el servidor puede enviar una imagen del sistema operativo por medios convencionales.

La ventaja principal de BOOTP consiste en poder bootear un host con un servidor que no se encuentra en la red local. Para eso existen dispositivos llamados BOOTP relay. Un relay es un dispositivo que puede recibir pedidos BOOTP pero que no tiene suficientes datos como para resolverlos solo. Cuando un relay recibe un pedido BOOTP, lo reenvía a un servidor de manera unicast. Puesto que se trata de un paquete IP válido, este puede viajar libremente por distintas subredes hasta llegar al servidor. El servidor completa los datos y se lo devuelve al relay, que inyecta la respuesta en la red. Para ayudar al servidor, el relay agrega su propia dirección IP en un campo especialmente diseñado para ello.

BOOTP no permite la asignación de datos dinámicos, estando limitado de una forma similar a RARP.

DHCP

BOOTP es un protocolo extensible, al final del formulario tiene un lugar asignado para extensiones. Una extensión de BOOTP es DHCP(RFC-2131). Esto le permite heredar funcionalidad de BOOTP, al mismo tiempo que tiene capacidades mucho mayores. DHCP trabaja sobre UDP/67 y UDP/68, y puede ser transportado por los BOOTP relays. Los servidores DHCP también pueden servir a clientes BOOTP. A pesar de ser una extensión, la dinámica del protocolo es tan diferente que se los pueden considerar protocolos distintos.

DHCP utiliza el área de extensión para definir sus propios mensajes. Aparte de poder indicar una dirección IP, DHCP puede indicar muchos otros parámetros, como máscara de red, servidores DNS y WINS, servidor de sincronización de hora, y ruta default, entre otros parámetros. DHCP también soporta la asignación dinámica de direcciones, incluyendo tiempo de asignación y asignación desde un pool.

DHCP define una serie de mensajes a enviar. A diferencia de BOOTP, la negociación no es en una sola etapa, sino que se negocian una serie de parámetros y luego se renuevan a lo largo de la sesión.

Inicialmente un cliente envía un mensaje DHCPDISCOVER al broadcast local. Este mensaje sirve para descubrir servidores DHCP que puedan proveer direcciones. Cada servidor que escuche ese mensaje (directamente o retransmitido por un relay) responde con un DHCPOFFER. Este mensaje contiene una propuesta para el cliente. El cliente luego elige entre las distintas propuestas y envía al broadcast local un mensaje DHCPREQUEST. Ese mensaje sirve para avisar al servidor correspondiente que el cliente acepta una propuesta, al mismo tiempo que avisa a los otros servidores que rechaza las propuestas que no se encuentran en ese mensaje. Si todo sale bien, el servidor responderá con un DHCPACK, indicando que está de acuerdo con el cliente. En cualquier momento, si hubiera un problema, el servidor puede responder con un DHCPNAK, indicando que hay un problema, y forzando al cliente a reiniciar la búsqueda. Esto puede ocurrir porque el servidor está obligado a asignar la dirección al recibir el DHCPREQUEST. Es posible que esa dirección haya sido asignada a otro host mientras se completaba la transacción.

En ese intercambio se negocian los parámetros de red (que el cliente pidió en el DHCPDISCOVER y DHCPREQUEST), y el período de validez de los datos. A diferencia de BOOTP, el servidor presta las direcciones con un límite de tiempo. Esto se conoce como un “lease”. Un cliente usa un mismo número de identificación a lo largo de un lease. Un lease debe ser renovado periodicamente, o expira. La renovación se realiza repitiendo el mensaje de DHCPREQUEST, al que el servidor vuelve a contestar con un DHCPACK. El lease puede ser terminado voluntariamente por el cliente, enviando un mensaje DHCPRELEASE.

Al negociar un tiempo de lease, también se negocian otros dos timers, T1 y T2. T1 es el tiempo en el cual el cliente debe empezar a renovar el lease con el servidor que se lo otorga. La renovación es un simple DHCPREQUEST unicast que debe ser respondido con un DHCPACK por el servidor. Si por alguna razón el servidor no responde y se excede T2, el cliente debe reiniciar el protocolo completo, buscando renovar su dirección desde cualquier otro servidor. Para esto manda un DHCPREQUEST broadcast. Si el lease se vence, el cliente debe descartar su configuración y reiniciar el protocolo desde el principio.

Una aclaración importante es que no hay un mecanismo estandar de sicronización de pooles de direcciones. Algunos servidores poseen mecanismos propios, pero estos mecanismos deben garantizar que no se solapen las direcciones asignadas. Es práctica común asignar pooles disjuntos en distintos servidores para proveer redundancia. También es posible hacer asignaciones estáticas con DHCP, aparte de las dinámicas.

Tunneling

Introducción

Suponga el siguiente caso:

(Gráfico: dos LAN (192.168.0.0/24) con tres hosts cada una (.2, .3 y .4), dos routers (marcados como [192.168.0.1, 200.123.4.2] y [192.168.1.1, 202.205.109.208]), una nube (Internet). Los routers conectan la nube con sus respectivas LAN.)

Se tienen dos LANs con direcciones privadas. Estas LAN tienen cada una un router conectado a Internet. Se desearía conectar las dos LAN entre si, pero no hay un enlace dedicado (y las direcciones privadas no se pueden usar en Internet). Normalmente estas dos redes no podrían conectarse una con la otra. Afortunadamente, estas redes se podrán conectar si los dos routers cooperan.

Caso básico: IP/IP

Supongamos el host 192.168.0.2 envía un paquete IP a 192.168.1.3, y que los dos routers pueden tener tablas de ruteo más complicadas de lo normal y que el router conectado a 192.168.0.0/24 tenga una entrada que diga:

Destino	interfaz	Next hop	Extras
192.168.1.0/24	----	202.205.109.208	Encapsular el paquete IP en IP, dirigido al NH

Mientras que el router opuesto tiene una tabla similar, dirigiendo el tráfico a 192.168.0.0/24 a 200.123.4.2 usando IP/IP.

Entonces el paquete resultante sería:

N2	IP(1) SRC:200.123.4.2 DST:202.205.109.208 Protocolo: IP/IP	IP(2) SRC:192.168.0.2 DST:192.168.1.3	Datos	CRC N2

Al llegar este paquete al router 202.205.109.208, este ve que el paquete está dirigido a él y lo acepta. Viendo que se trata de IP sobre IP, desencapsula el paquete interior, obteniendo un paquete IP dirigido a la LAN privada que tiene directamente conectado. Siguiendo las instrucciones de su tabla de ruteo, lo entrega a 192.168.1.3 que está directamente conectado.

El encapsulamiento puede verse de dos formas distintas. El paquete tiene dos niveles 3, uno encapsulado dentro de otro. Entonces si analizamos este paquete del nivel de red, hay dos posibles interpretaciones. La primera interpretación es considerar a IP(1) como el nivel 3. En ese caso, IP(2) es parte de los datos e irrelevante para el ruteo. Si se ve a IP(2) como el nivel 3, el encabezado IP(1) puede ser considerado como parte del nivel 2.

N2	N2	IP(2) SRC:192.168.0.2 DST:192.168.1.3	Datos	N2
N2	IP(1) SRC:200.123.4.2 DST:202.205.109.208 Protocolo: IP/IP	Datos	Datos	N2

Dependiendo de la situación, podemos usar las dos interpretaciones.

Si el paquete es analizado en tránsito mientras pasa por la Internet, el encabezado IP(1) está sobre el nivel de enlace. Para la Internet, ese es el nivel 3: el paquete es un simple paquete IP que transita de un router a otro. Los datos son irrelevantes. Puesto que los dos routers tienen direcciones válidas, el paquete puede transitar sin problemas.

Si analizamos el paquete a medida que pasa por cada salto, podemos usar la interpretación alternativa. Al salir de 192.168.0.2, el encabezado IP(2) es el encabezado de red. El host busca el destino de IP(2) en la tabla de ruteo y actúa de acuerdo a la información de su tabla. Cuando el paquete es entregado al router de salida, IP(2) sigue siendo el nivel de red. El router actúa en base a la dirección de destino y a su tabla de ruteo. Al TTL de IP(2) se le resta uno. En este momento se inicia el tunneling. Se agrega el otro encabezado IP y el paquete se despacha por Internet. El encabezado IP(2) es parte de los datos y queda congelado. A pesar de que está pasando por varios saltos, el encabezado de IP(2) no es alterado. El paquete visto desde el punto de IP(2) está en un túnel y se comporta como si estuviera en un cable largo. Cuando el paquete llega al router 202.205.109.208 y es desencapsulado, el router vuelve a tratar a IP(2) como el nivel de red, busca el destino en su tabla de ruteo y le resta uno al TTL. Luego lo envía. Para IP(2), existieron solamente tres saltos, aunque uno de esos saltos haya sido un viaje por Internet, desde el punto de vista de IP(2) fue simplemente el tránsito por un enlace (particularmente largo).

El resultado práctico de este tunneling es que las dos redes se pueden comunicar a pesar de tener direcciones privadas y estar separadas por Internet. Esto se debe a que para esas redes, la seguda interpretación es la válida, ya que no les importa que un par de routers en el medio creen un túnel. Para los hosts en ambas redes, hay un enlace directo entre ambos routers. El hecho de que este enlace sea virtual no hace diferencia alguna.

Casos avanzados

IP/IP no es el único caso de túnel posible. Practicamente cualquier protocolo de nivel 3 se puede encapsular en casi cualquier otro. Un ejemplo sería encapsular IP en X.25 (un protocolo de circuito virtual). Esto le permitiría a dos redes IP comunicarse usando una red incompatible (la red X.25). También es posible encapsular otros protocolos, como protocolos de nivel de enlace sobre un protocolo de nivel de red. Un ejemplo de esto sería encapsular ethernet sobre IP. Imaginemos que en la situación anterior quisieramos que ambas redes formaran parte de la misma LAN. Entonces en vez de encapsular IP en IP usando un router, podríamos haber instalado un dispositivo híbrido que cumpla las funciones de switch hacia el lado de la LAN y router hacia Internet. El switch híbrido estaría comunicandose con un dispositivo similar conectado a la otra LAN, como si se tratara de dos switches con un cable conectándolos, solo que en lugar de un cable encapsularíalos frames ethernet en IP. Un mensaje enviado en una de las LAN sería capturado y enviado a la otra gracias a esos dispositivos híbridos, y ambas redes estarían fusionadas como si estuvieran unidas por dos switches. Otras variantes también son posibles, como por ejemplo encapsular 802.1q en 802.1q. Esto haría una especie de trunk de trunks. Esto permitiría que una entidad distribuyera por medio de una red ethernet el tráfico de otras redes ethernet (clientes), sin que estos se interfieran mutuamente o siquiera sospechen que están compartiendo una única red de distribución. Para los clientes, la red de distribución parecería dedicada a ellos.

Es posible también encapsular distintos protocolos sobre protocolos de nivel superior al de red. Por ejemplo se puede encapsular IP sobre TCP para asegurar un paso confiable por cierto tramo (cosa no recomendada por razones que exceden el alcance de este documento). También es posible encapsular IP o TCP sobre HTTP, a fin de evadir ciertos controles. Para ello es necesario tener un “cómplice” más allá del control que desempaquete el tráfico (en esquemas de túnel, siempre es necesario que hayan dos partes, debe mantenerse la simetría). El protocolo SSH (a ver al final del cuatrimestre) también provee su propia forma de encapsular TCP.

Las combinaciones de encapsulamiento son muy variadas y es posible realizar múltiples encapsulamientos de acuerdo a la necesidad de cruzar distintas redes.

VPN

En los casos mencionados anteriormente, no hay ninguna protección de los datos. A pesar de que normalmente el contenido del paquete que encapsula el tráfico debería ser ignorado, no hay nada que impida que un nodo hostil analice el contenido de los mensajes, determine los orígenes y destinos internos y copie o altere los datos. Para evitar este tipo de intrusiones o análisis, se utilizan las llamadas redes privadas virtuales (VPN). Una VPN es simplemente un túnel, con una diferencia: en lugar de encapsular directamente, a los datos encapsulados se los protege por medios criptográficos. Los routers mencionados en los ejemplos anteriores poseen claves y antes de encapsular los datos los cifran, agregan headers de protección y toman otras medidas. Esto se lo puede ver como un encapsulamiento en un protocolo de usuario que realiza las tareas de seguridad, que luego es montado sobre un protocolo de red. Esto causa que los datos sean opacos para alguien que no posea la clave correspondiente.

N2	IP(1) SRC:200.123.4.2 DST:202.205.109.208	HEADER VPN	PROTEGIDO IP(2) invisible para afuera	PROTEGIDO Datos invisibles para afuera	CRC N2

Un nodo intermedio que viera un mensaje protegido no podría saber más que se trata de un mensaje entre esos dos routers especiales. Idealmente los campos protegidos por la VPN también tiene mecanismos para evitar su alteración.

NAT

Introducción

A veces una red puede no tener direcciones públicas para todos sus nodos. Una red puede tener asignadas direcciones privadas, pero querer conectarse con hosts externos, lo que requiere una dirección pública. Normalmente esto no sería posible, aunque gracias a la técnica de NAT esto se puede lograr (aunque no sin dificultades).

En este documento usaremos el término “NAT” para la traducción a nivel 3, y dejaremos el término “PAT” para la versión de esta técnica realizada a nivel de transporte (también conocida como NAT, o masquerading)

NAT

NAT (Network Address Translation) consiste en traducir la dirección de origen de un paquete, reemplazándola por una dirección pública. A diferencia de tunneling, esta técnica es unilateral. No hay cooperación del otro lado, por lo que no es posible encapsular. Esto se debe a que no se desea atravesar una red para luego llegar a otra, sino que se desea establecer una comunicación con la red incompatible.

Para lograr esta comunicación, es necesario tener un router especial. Este router debe tener asignado un pool de direcciones públicas. Los hosts internos tienen direcciones privadas. Estos hosts deben tener su tabla de ruteo configurada para que su tráfico salga por ese router especial. Si por alguna razón parte de su tráfico no pasara por ese dispositivo, la traducción fallaría.

Ese router va a tener una tabla, en donde se pueden guardar direcciones de origen, destino y direcciones de traducción. Cuando ese dispositivo recibe tráfico de un host interno, debe anotar en la tabla la dirección de origen y destino. Luego debe tomar una dirección del pool (que no esté asignado a otro host, al menos con ese destino) y se la asigna al origen. Ese dato lo anota en la tabla junto a los otros datos. Una vez hecho esto, el router debe reemplazar la dirección origen (que es privada) por la dirección que asignó del pool (que es pública). Luego despacha ese paquete al

destino. Si recibe más paquetes del mismo origen al mismo destino, los debe traducir de acuerdo a los datos de la tabla.

Cuando el destino responde, este envía las respuestas a una de las direcciones del pool (la que fue asignada a esa comunicación en la tabla). Estas respuestas son ruteadas a ese router especial. Una vez en el router, este debe mirar la dirección de destino (que es una dirección del pool) y la de origen (que está en la tabla como un destino). Si hay una coincidencia, entonces ese paquete debe ser redirigido al host indicado como de origen.

Despues de realizar la traducción, el host de origen y el destino ven dos cosas distintas, pero ambas son aceptables desde su punto de vista, por lo que la comunicación tiene éxito. Para el host de origen parece haber una comunicación normal. A pesar de que tiene una dirección privada, se comunica con éxito. El host destino ve una comunicación viniendo de una dirección pública, lo que no es un problema.

Problemas con NAT

La traducción no siempre es exitosa. El primer problema de NAT es que requiere un pool de direcciones. Si no llegaran a haber suficientes direcciones disponibles, la comunicación no podrá ser concretada.

El principal problema de NAT es la incompatibilidad con ciertos protocolos. Puesto que se está realizando una traducción, el host de origen y el destino no ven la misma información en los encabezados. Normalmente esto no sería un problema, ya que en teoría las capas son independientes, pero en la práctica esto no es así. Si un protocolo de nivel superior usara algo del nivel de red como identificador (por ejemplo usa la dirección del host) en un campo del nivel superior, este no será traducido y se generará una discrepancia. Si el nivel superior realiza una operación de red en base a esa dirección, la comunicación no tendrá éxito por tratarse de una dirección privada. Adicionalmente varios hosts podrían presentar la misma dirección privada, confundiendo al servidor.

Este problema se debe a que ese identificador en el nivel superior no es traducido. Una posible solución consiste en programar al router híbrido para que analice el protocolo de nivel superior y realice una traducción en el nivel superior, pero esto requeriría mucho más poder de cómputo y conocer todos los posibles protocolos superiores, lo que no siempre es posible.

Un problema adicional aparece cuando se deben establecer múltiples conexiones en varios sentidos. En el caso de NAT puro, solamente se manifiesta si un segundo host se debe comunicar con el origen. Esto rara vez es un problema en este tipo de traducción, pero la ocurrecia de este tipo de eventos se multiplica a medida que la traducción se hace en niveles superiores (es un problema a nivel 4, la traducción más común, puesto que este error puede cometerse aún con tráfico proveniente de la misma dirección de red).

PAT

NAT puro no es muy común, debido a que se necesita un pool de direcciones. Una variante mucho más usada es PAT, que realiza la traducción en el nivel de transporte. En este caso basta con tener una única dirección pública, ya que el nivel de transporte tiene la capacidad de multiplexar, lo que nos permite crear un pool de relaciones de nivel 4 sin recurrir a múltiples direcciones de nivel 3. Este tipo de traducción es la que sucede en un dispositivo como un router hogareño que usa TCP/IP.