Administración de Sistemas y Servicios Telemáticos

Rsync

Víctor Abeytua García
50867285 - W

Departamento de Ingeniería de Sistemas Telemáticos

Profesores: Joaquín Seoane Pascual
Tomás P. de Miguel

1) Objetivos y descripción del servicio

Rsync es un programa que viene a sustituir al rcp (programa para copiar ficheros desde/a máquinas remotas), que además incorpora muchas mejoras. Es un programa que en su concepción más básica sirve para copiar un fichero de un sitio del disco duro a otro e incluso desde un ordenador a otro (sin que tengamos que estar físicamente nosotros ante ninguno de los dos), pero no es ahí donde le podemos sacar el mayor partido a este programa ya que en esas situaciones no tiene ninguna ventaja con respecto a programas como el cp (programa de copia por defecto en entornos UNIX) o el anteriormente mencionado rcp. Donde de verdad demuestra su valía es cuando disponemos de una copia del fichero y lo que queremos es actualizarlo con otro similar (queremos sincronizar ambos ficheros). Si el fichero no es muy grande el procedimiento habitual sería copiar el fichero nuevo y borrar el antiguo. Pero ¿qué pasa cuando el fichero que tenemos que actualizar es de 600 Mb? Si los dos ficheros se encuentran dentro del mismo ordenador, o incluso forzando la situación se encuentran conectados a través de un enlace muy rápido (y ahí entrarían en juego otros factores como la saturación de la red, límite de tamaños etc ...) podríamos seguir usando los programas y el procedimiento antes mencionado, pero cuando nunca podríamos hacerlo sería con dos ordenadores conectados entre sí a través de un enlace lento (como pueden ser conexiones a través de módems de 56K etc ...). Transferir esa cantidad de información a través de ese tipos de enlaces podría llevar fácilmente ¡ 2 días ¡ y eso contando con que no se corte la conexión en ningún momento y que nos podamos permitir tener la conexión durante tanto tiempo abierta. Es aquí, en este tipos de casos (además de los otros ya que sería una forma de no malgastar recursos) donde el rsync, utilizando el "algoritmo rsync" obtiene sus mejores rendimientos. Porque el rsync, a diferencia de otros métodos, no necesita transferir todo el fichero para comprobar si los dos son iguales, es capaz a través de una serie de checksums (generados a partir del fichero que se quiere actualizar, enviados a la ubicación del otro fichero y recalculados sobre él) de hallar las diferencias entre los ficheros (si las hubiera) y de enviar sólo esas diferencias. Esto traducido en números podría repercutir que en la actualización del fichero de 600 Mb, en vez de transferir toda esa cantidad de información necesitáramos transferir unas 200 Kb (con lo que el aumento en el gasto de computación está más que justificado).

Mencionar, que otra de las funcionalidades del rsync es la de actuar como servidor de rsync, a donde usuarios anónimos se pueden conectar para actualizar ficheros (como pueden ser imágenes de cd's de distribuciones de linux etc ...).

Por último sólo queda resaltar cuales son las formas de ejecutar el programa:

rsync [opción] fichero_origen [usuario@]host:destino
(Sincronizar un fichero en otra máquina con uno en local)

rsync [opción] [usuario@]host:fichero_origen destino
(Sincronizar un fichero en local con uno en otro máquina)

rsync [opción] fichero_origen destino
(Sincronizar un fichero en local con uno en local)

rsync [opción] [usuario@]host::fichero_origen [destino]
(Sincronizar un fichero local con uno en un servidor rsync)

rsync [opción] fichero_origen [usuario@]host::destino
(Sincronizar un fichero en un servidor rsync con un local)

rsync [opción] rsync://[user@]host[:port]/fichero_origen destino
(Sincronizar un fichero local con uno en un servidor rsync especificando el puerto por el que accedemos al servidor)

2) Requisitos

A la hora de funcionar el rsync necesita estar instalado tanto en el sistema origen como en el sistema destino. Además usa el rsh o el ssh para comunicarse con otras máquinas, por lo que requiere que alguno de los dos esté instalado (otra vez tanto en el sistema origen como el destino). Evidentemente es recomendable que esté instalado el ssh ya que es usa encriptación en las comunicaciones con lo que se aumenta la seguridad frente a posibles "escuchas" protegiendo así las contraseñas que de otra manera se enviarían en "abierto".

Para que el rsync funcione (a no ser que queramos que nuestra máquina funcione como un servidor de rsync) no se necesita ni ser setuid ni privilegios especiales para la instalación. Tampoco necesita funcionar ni como demonio ni como parte del demonio inetd. Lo que sí necesita es estar en el "path" de ejecutables para que se pueda llamar cuando la sincronización se hace desde una máquina remota.

Por el contrario, si queremos que funcione como servidor, deberá funcionar como demonio con privilegios de súper usuario (root).

3) Arquitectura del servicio

Como hemos dicho antes, la idea básica del programa rsync es permitir sincronizar dos ficheros similares ubicados en distintas máquinas sin tener que transferir uno de los dos ficheros entero. Esto lo consigue el rsync calculando unos checksums sobre el fichero que se quiere actualizar, transmitiendo los checksums a la otra máquina, esta máquina (que ha de tener instalado también el rsync) calcula los mismos checksums sobre el fichero al que se quiere actualizar, los comprueba con los enviados y envía a la primera máquina las diferencias (si las hubiera) entre los dos ficheros y donde situarlas. Esto es lo que se conoce como el "algoritmo del rsync" y para poder estudiarlo de una forma más detallada lo mejor es verlo sobre un ejemplo.

Imaginemos dos ordenadores (ambos tienen instalado el rsync): α y β. α tiene acceso al fichero A y β tiene acceso al fichero B. Entre los dos hay un enlace de comunicaciones lento y se quiere que el fichero B sea idéntico al A.

Una vez tenemos este escenario pasamos a describir paso a paso como funciona el "algoritmo del rsync":

1) β divide el fichero B en bloques, que forman una partición (es decir, la unión de todos es el fichero y la intersección de cualquiera de los bloques es el conjunto vacío), de tamaño S bytes (aunque el último de los bloques puede no llegar a estos S bytes).
2) Sobre cada uno de estos bloques β calcula dos checksum: un checksum cíclico "débil" de 32 bits y uno MD4 fuerte de 128 bits.

3) β envía los checksums a α.
4) α busca por todo A todos los bloques de S bytes (no solo a offsets de S bytes sino a cualquiera) que tengan la misma pareja de checksums que alguno de los bloques de B. Esto se puede hacer de una sola pasada aprovechando una de las propiedades del checksum cíclico de 32 bits.
5) α le envía a β una secuencia de instrucciones para construir una réplica de A en β. Estas instrucciones pueden ser o una referencia a uno de los bloques de B o datos literales. Estos datos literales sólo se envían si había bloques en A que no estaban en B.

Una vez que hemos descrito el algoritmo, lo que nos falta por hacer es describir de una forma mucho más precisa la forma en la que se comprueban los checksums ya que es gracias a esa comprobación por la que es tan efectivo el funcionamiento del rsync.

Para describirlo supondremos el escenario anterior una vez α ha recibido los checksums generados por β a partir de B. La estrategia básica consiste en generar los checksums de 32 bits para todos los posibles bloques de S bytes de A hasta que se encuentre una coincidencia con los enviados de B (calculamos un bloque, si no hay coincidencia avanzamos un byte y volvemos a calcular etc.). Toda esta estrategia de comprobación se divide en tres niveles:

1. En el primer nivel consiste en crear una tabla hash con 2^{16} entradas. Esta tabla se rellena con los checksums de 32 bits de B siendo la clave de acceso a las posiciones de la tabla un hash de 16 bits de dichos checksums. Por tanto cada entrada en la tabla tendrá un checksum (ó varios, ya que, aunque no es fácil, el hash de dos checksums distintos puede ser el mismo) ó, en caso de que no haya ningún checksum que genere ese hash, estará vacía. Una vez generada esta tabla se calcula, a partir de todo offset, sobre A el checksum de 32 bits y su hash de 16 y se busca en la tabla. Si hay una coincidencia del hash se pasa al siguiente nivel.
2. En este segundo nivel lo que se hace es comprobar si existe en la tabla un checksum de 32 bits que coincida con el que hemos calculado. Para ello lo que hacemos es partir de la posición que nos ha dado el hash de 16 bits y recorrer la tabla hasta que encontremos ó bien una coincidencia, con lo que pasaríamos al siguiente nivel, ó bien una posición vacía ó una posición cuyo hash sea distinto .
3. Cuando se llega a este nivel se calcula el checksum de 128 bits. En caso de coincidencia se considera que hemos encontrado un bloque en A que es exactamente igual a otro en B (aunque en la práctica se podría dar el caso que los bloques fueran distintos, después de las dos comprobaciones que se han realizado, la probabilidad de error es tan pequeña que es completamente asumible), en caso contrario se vuelve al nivel inferior y se sigue buscando por la tabla hasta que encontremos una posición vacía o una cuyo hash es distinto.
   Cuando hay una coincidencia α le envía a β la información de A que hay desde la última coincidencia (si fuera la primera, desde el principio) hasta el inicio del bloque que ha coincidido, seguido del índice del bloque que ha coincidido. Esta información se envía inmediatamente después de una coincidencia para que se pueda superponer computaciones sucesivas. Después de enviar los datos, la computación se reanuda desde el final del bloque que ha coincidido.

Por último (en cuanto al funcionamiento del algoritmo), hay que mencionar que rsync permite pipelining (encadenamiento). Esto quiere decir que si se tienen que actualizar/sincronizar varios ficheros se podría funcionar de la siguiente manera para minimizar el efecto de la latencia mientras la máquina que tiene los ficheros comprueba los checksums. Esta manera implica que β lance dos procesos independientes, uno de ellos procesa los ficheros y envía los checksums, mientras que el otro recibe las diferencias que envía α y reconstruye los ficheros.

Finalmente hay que mencionar que como el rsync utiliza o bien ssh o rsh para comunicarse con otros ordenadores quiere decir que se sitúa en el nivel de aplicación por encima del protocolo TCP. Ya que tanto ssh como rsh son dos servicios orientados a conexión (algo que no soporta UDP) y por tanto tienen que usar ese protocolo.

4) Configuración

El rsync, al ser un programa que se ejecuta invocado desde la línea de comandos y no como un demonio, no requiere ningún tipo de configuración especial. No necesita ni ser setuid ni requiere ningún tipo de privilegio especial a la hora de instalarse. Lo único que requiere es, para que se pueda invocar remotamente, que esté situado en la ruta de directorios.

En cambio, si lo que se quiere es que el rsync funcione como servidor tenemos que ejecutar el rsync como demonio con privilegios de root:

rsync --daemon

Con esta opción el rsync escucha en el puerto TCP 873 y acepta conexiones tanto autentificadas como anónimas para sesiones de rsync.

El servidor tiene un fichero de configuración llamado rsyncd.conf y situado en el /etc. Este fichero controla la autenticación, tipos de acceso, donde se guardan los "logs" del sistema y los módulos disponibles.

La estructura de este fichero de configuración es la siguiente:

1) Está basada en comandos de una sola línea (el comienzo de línea indica un nuevo nombre de módulo, un comentario o un parámetro).
2) Un módulo empieza con el nombre del módulo dentro de [] y continua hasta que empieza el siguiente módulo. Contiene parámetros del tipo: 'name = value'
3) Toda línea que empiece por un # es un comentario.
4) Toda línea que acaba en una \ continua en la siguiente (como es común en todos los sistemas Unix).
5) En los módulos se puede configurar parámetros como: path, número máximo de conexiones, sesiones de sólo lectura, autenticación de usuarios etc.

5) Pruebas

Para probar el rsync lo que he hecho ha sido seguir paso a paso las instrucciones que se encuentran en http://www.debian.org para conseguir y crear las imágenes ISO (formato para grabar CD's) de la distribución oficial de Debian.

Debido al gran éxito que tiene esta distribución es físicamente imposible que todo el mundo que quiera crearse su propio CD de Debian se conecte a su página web y se baje los cerca de 1.2 Gb que ocupan los CD's de la distribución oficial. Por ello se optó por un método muy habitual con todo tipo de programas de éxito que se difunden por Internet: tener distintos servidores a lo largo del mundo con copias de los ficheros de donde los usuarios anónimos los pueden descargar (lo que en inglés se conoce como mirroring). Pero debido a que una distribución no es del todo estática (a diferencia de programas binarios, que se quedan tal y como se distribuyen hasta que haya una nueva versión), y se van produciendo, casi de forma constante, pequeños cambios (se corrigen bugs, se añaden/quitan trozos de código) en la misma esta solución no es del todo eficaz. Por tanto se llego a una solución que permite tener la web de Debian accesible (no tiene una carga tan alta al no tener que estar descargando tantas veces ficheros tan grandes) y a los usuarios finales conseguir las imágenes ISO completamente actualizadas.

La solución consiste en que toda persona que quiera conseguir las imágenes ISO de los CD's de la distribución de Debian (y no se identifique como mirror oficial de Debian) las tendrá que descargar de un mirror oficial de Debian. Como he dicho antes, estas imágenes no van a estar del todo actualizadas, pero es aquí donde entra en funcionamiento el rsync. Una vez que contamos con las dos imágenes en nuestro ordenador nos conectamos al servidor rsync de Debian, para que el rsync halle las diferencias entre ambos ficheros (recordemos que cada uno ocupa aproximadamente 600 Mb) y descargue las diferencias.

Es cierto que en esta prueba nos hemos tenido que descargar a pesar de todo los cerca de 1.2 Gb que ocupa la distribución, pero bueno, de algún sitio teníamos que conseguir unos ficheros así, y lo que es más importante, que luego podamos encontrar esos mismos ficheros pero con pequeñas diferencias en otros sitios de Internet. Pero si omitimos esta descarga inicial, nos damos cuenta de la efectividad del rsync.

Los datos que obtuve son aproximados, ya que la red en la que me encontraba se usa bastante. Desde que inicié la sincronización de los dos ficheros el trasvase de información entre los dos ordenadores fue menor a 1 Mb. Si tenemos en cuenta el tamaño total de los dos ficheros, se puede apreciar claramente lo efectivo que resulta el rsync para este tipo de usos.

La instalación del servidor de rsync radica únicamente en ejecutar el programa como demonio. Una vez que empieza a funcionar el solito se encarga de gestionar las conexiones de usuarios que quieren actualizar sus ficheros.

6) Gestión diaria

El programa en sí no requiere ningún tipo de gestión diaria, ya que simplemente consiste en un ejecutable. Evidentemente lo que si que hay que supervisar es que la conexión a la red este funcionando, pero eso es algo que no siempre depende de nosotros, por lo que no se puede considerar como gestión propia del rsync.

En cambio si lo que queremos tener es un servidor de rsync en nuestro ordenador hay una serie de pasos que hay que cumplir. Una cosa fundamental (al igual que con cualquier tipo de servidor que estemos gestionando), es comprobar si en los logs (la ubicación de este fichero se especifica en el fichero de configuración: rsyncd.conf) del programa aparecen mensajes de error. Estos mensajes de error nos pueden indicar que hay usuarios que solicitan ficheros que no existen (y que a lo mejor debieran) o que no tienen los permisos adecuados para que se puedan descargar. También nos pueden indicar que hay gente intentando acceder como usuario anónimo y habíamos deshabilitado esa opción ó con distintos nombres de usuarios erróneos de una forma muy seguida (lo que podría indicar la existencia de un intento de ataque).

Otra cosa que se tiene que hacer siempre (y en especial con servidores que escuchan es puertos TCP/IP y que por tanto pueden ser vulnerables a ataque por red) si se es administrador, y por tanto nos vale para este caso, es mantenerse informado de posibles bugs que se descubran tanto en el programa en cuestión como en todo lo referente a la pila de protocolos IP.

Por tanto el rsync requiere una gestión diaria propia mínima, que se debería englobar más como parte de las tareas del administrador del sistema. Ya que los mayores problemas que pueden surgir forman parte de las tareas habituales de dicha persona.