

NOTICIAS DEL KERNEL

LISTOS PARA EXT4

En un masivo flameware que abarcó centenares de posts a la lista de correos, los desarrolladores del kernel dirimieron sus diferencias sobre el plan a seguir en el desarrollo del sistema de ficheros ext3. El elemento catalizador de la trifulca fue un envío de Mingming Cao, donde anunciaba un esfuerzo por parte de Red Hat, ClusterFS, IBM y Bull de convertir ext3 de un sistema de ficheros de 32 bits, a uno de 48 bits, permitiendo incrementar el tamaño del sistema de ficheros desde un máximo de 8 terabytes a un máximo de 1024 petabytes.

A lo largo de la discusión, se reveló que otros desarrolladores también estaban trabajando en la implementación de sus propias extensiones para ext3. Las extensiones de sistemas de ficheros son una manera de mantener los datos de un disco contiguo en vez de dejar que se extiendan a lo largo de bloques dispares.

Todos estos cambios de bajo nivel que provienen de varias fuentes han dado pie a la pregunta de si es apropiado asignar este trabajo al proyecto ext3, o si es hora de empezar una nueva versión y denominarla ext4.

Por un lado, todas las propuestas son características potencialmente positivas y útiles. Los usuarios se beneficiarían y, si se implementan correctamente, tendrían un impacto en la velocidad y el tamaño, dos problemas que importan mucho cuando hablamos de sistemas de ficheros.

Por otro lado, ext3 se convirtió en monstruo, hinchado, difícil de mantener y difícil de arreglar, encadenado a características cuestionables, como ínodos grandes que sólo son de utilidad a un pequeño porcentaje de usuarios con datasets enormemente grandes. Apilar nuevas características encima de esta, ya de por sí, estructura inestable puede que sólo empeore los problemas.

Linus dice que favorece la idea de ext4, lo que ha desencadenado una masiva cantidad de protestas desde muchos sitios. A primera vista, no parece que haya tanto en juego. Linus ha comentado en más de una ocasión que no tiene nada en contra de ninguna característica específica de ext3, el desarrollo seguiría sus actuales derroteros, todo es igual, excepto por el hecho de que los desarrolladores copiarían su código a un nuevo árbol "ext4" y continuarían

su desarrollo desde ahí. ¿A qué tanto lloro?

El primer problema es que no hay en estos momentos ningún usuario de ext4 por ahí fuera para probar el código. Una vez ocurra el fork, los desarrolladores de ext4 perderán toda su base de usuarios, que en estos momentos abarca millones en todo el mundo.

Otro problema es portar todas las mejoras de ext4 y los arreglos a sus fallos de vuelta a ext3 en el momento idóneo. ¿Cómo se piensa enfocar? ¿Y por qué cargarse con tanto trabajo? ¿Y no significaría esto una divergencia inevitable entre dos bases de código muy similares? ¿Cómo se puede mantener el código compartido entre ext3 y ext4 sin que se haga infinitamente complejo?

El hecho es que parece que todos los mantenedores de ext3 se oponían al fork, por tanto ¿por qué molestarse?

La respuesta de Linux es similar a las cosas que ha dicho con anterioridad sobre otros proyectos y tiene que ver con el cambio básico de cómo se visualiza un proyecto y su código fuente. De hecho, su respuesta consiste en deshacerse de las preguntas. El portado inverso no es necesario, dice, los bugs más importantes se arreglarán en ext3 y a la mayoría de los usuarios de los sistemas de ficheros no les importará si tienen lo último de lo último en prestaciones o no. ■

POLÍTICAS PARA EL FORMATO DE PARCHES

Eric W. Biederman intentó modificar el sistema de administración de git para permitir que el verdadero autor de un parche pudiera incluir su nombre de una manera más relajada. Por ejemplo, este cambio en git permitiría al autor indicar su nombre dentro del cuerpo del correo

La lista de correo del kernel de Linux comprende lo principal de las actividades de desarrollo de Linux. El volumen del tráfico es inmenso, alcanzándose a menudo los diez mil mensajes semanales. Mantenerse al día de todo lo que sucede en el desarrollo del kernel es casi imposible para una sola persona.

Sin embargo Zack Brown es uno de los pocos valientes que lo intentan y a partir de ahora, podrá leerse lo último de las discusiones y decisiones con respecto del kernel de Linux llevados de la mano de este experto.

Zack ha publicado un resumen online semanal llamado "The Kernel Traffic Newsletter" durante cinco años. Linux Magazine te trae ahora la quintaesencia de las actividades del kernel de Linux del mayor especialista en el tema.



electrónico en vez de exclusivamente en la parte superior del mensaje. Pero Linus Torvalds ha sido expeditivo en su negativa a admitir el cambio. “Desde el principio del desarrollo de git,” dijo Linus, “he intentado ser muy claro de que no debe haber nunca un juego de adivinanzas en marcha. No utilizamos la ‘heurística’ excepto como herramienta de optimización, es decir, la heurística puede tener un impacto en el *rendimiento*, pero nunca, jamás de los jamases debe tener un impacto semántico.”

Siguió diciendo: “si el nuevo git-applymbox toma líneas al azar desde el cuerpo del mensaje de correo y decide que puede ser información sobre la autoría, entonces es un BUG. El campo ‘From:’ en el medio del correo podría perfectamente indicar la persona que *descubrió* el bug y le citamos como parte de la explicación. No indica necesariamente la autoría.”

Eric estuvo de acuerdo en hacer las cosas como pedía Linus, pero estaba un poco molesto por el hecho de que un parche anterior suyo había conseguido burlar el radar de Linus y había entrado en el árbol de git, haciendo que tuviera que malgastar más tiempo en el trabajo actual. ■

LINUX Y EL IRC

Linus Torvalds le ha dicho a los desarrolladores que favoreciesen la lista de correo de git en git@vger.kernel.org sobre el canal de IRC. Dijo odiar el IRC, y señaló que la lista de correo tenía una política de envío abierta, lo que significa que cualquiera puede remitir sus preguntas, tanto si están suscritos, como si no. Y si la lista de correo de git sigue la tradición de la lista de correo del kernel, cualquiera que responda a la pregunta, incluirá copia al remitente original.

La de mantener una lista de desarrollo abierta es una tradición por la que se ha tenido que luchar. El que cualquiera pueda enviar su pregunta, implica también que cualquiera puede enviar su spam. En la mayor parte de los casos, los administradores de la lista de git, del kernel y de los centenares de otras

listas en los servidores “vger”, hacen una tarea increíblemente eficiente al evitar que el spam se filtre a las listas. Pero es una tarea dura, que requiere una vigilancia continua y no está exenta de fallos. Así que, de vez en cuando, alguien (o un grupo de “alguiens”) sugiere permitir el envío de mensajes sólo a los miembros suscritos. Pero los administradores de las listas y el mismo Linus son categóricos al respecto, alegando que cualquiera en todo el mundo debe de poder enviar sus informes de error y otras experiencias relacionadas con el kernel. Y lo mismo se aplica a git.

No todas las listas de Linux siguen la misma política. El spam es un problema tremendo y no todos los administradores de listas de correo tienen el tiempo de combatirlo, especialmente cuando existe una solución sencilla, como la del miembro suscrito. Y también es sabido que no todo el mundo coincide con Linus al respecto de cómo conducir una lista de correo. Debido a la diversidad del propio modelo de desarrollo de Linux, no sólo se pueden llevar a cabo esas ideas en la práctica, sino que pueden esperar un cambio de las preferencias de Linus en el caso de que el spam se desboque y se necesite realizar algún cambio. ■

LOS COLORES DEL KERNEL

Algunos estándares requieren que grandes comités oficiales los debatan durante años antes de llegar a un consenso. Otros estándares, por contra, se despachan con un par de mensajes a la lista de correos del kernel.

La decisión de si se estandarizaba la ortografía de “color” (o “colour”) como “colour” (o “colour”) lo resolvió David Woodhouse cuando Andrew Morton señaló que uno de sus parches conllevaba las dos ortografías en el mismo bloque de código.

Para arreglar el dilema, David escogió “color”, debido a que el código en cuestión utilizaba “color” en un API público, mientras que “colour” sólo aparecía como el nombre de una variable interna.

El interfaz más rígido salió triunfante en este caso. O, al menos triunfa hasta el momento. En el momento de escribir esto, existen centenares de instancias de “colour” en el código del kernel, frente a más de 2000 instancias de “color”.

“Colour” lo tiene bastante negro. ■

INODOS A REGIMEN

Theodore Y. T’so decidió hacerse cargo de la compactación del struct de inodos del kernel. Esto tendría un impacto no sólo en el uso de memoria de ext3, sino en la de todos los sistemas de ficheros, ya que el struct de inodos es compartido de forma común por todos. Ted descubrió con varios tipos de datos que se podrían extraer del struct sin causar mucho daño, pero admitió que “examinar todo el código que utiliza estas variables antes de extraerlas sería todo un desafío de programación”. Invitó a los otros desarrolladores a que le ayudasen, y un cierto número de ellos, incluyendo Alexander Viro, se unieron a su causa.

Diez días después, con el apoyo de Linus, publicó una serie de parches invasivos que obtenían un buen número de los objetivos que se había propuesto alcanzar. Sin embargo, no todos los cambios eran totalmente satisfactorios y un importante número de desarrolladores propusieron rectificaciones a nivel técnico, pero la respuesta general fue muy favorable. No es que saliesen a bailar por las calles, pero mucha más gente se comprometió a ayudar a mejorar los parches. ■

SOPORTE PARA BANDA-ULTRA-ANCHA

Inaky Pérez-González, en nombre de Intel, ha anunciado su proyecto para implementar soporte para hardware en desarrollo que sea compatible con la Wimedia Ultra Wide Band (UWB) y los estándares Wireless USB. UWB es una tecnología de redes wireless de corto alcance, optimizado para lugares cerrados. Si bien el hardware aún no está disponible, o es difícil de conseguir, Inaky invitó a la gente de Linux a unirse al proyecto para ayudar en el desarrollo. ■