Novedades en IBM Communications Server para Linux V6.2.1

Esta publicación describe los cambios de release de mantenimiento para la versión 6.2.1.0 de Communications Server para Linux (Núm. de producto 5724-i33, CS Linux) y de Communications Server para Linux sobre zSeries (Núm. de producto 5724-i34, CS Linux sobre zSeries). Las descripciones son aplicables a los dos productos, Communications Server para Linux y Communications Server para Linux sobre zSeries.

Los cambios proporcionados en este release de mantenimiento para Communications Server para Linux V6.2.1.0 incorporan:

1. Soporte de kernel 2.6 - Soporte de sistema operativo Linux actualizado
2. Linux sobre Power - Soporte de plataforma y empaquetamiento adicionales
3. AIX Remote API Client - Plataforma y empaquetamiento adicionales
4. Soporte de LU primaria - Interfaz de programación de LUA para soportar flujos de LU 0 primaria
5. Actualizaciones de Remote API Client - Actualizaciones proporcionadas para los clientes
6. Actualizaciones de la publicación LUA Programmer's Guide (guía del programador de LUA) - Interfaz de LU primaria adicional
7. Actualizaciones de la publicación Administration Guide (guía de administración) - Mandatos adicionales para definir y consultar parámetros del temporizador HPR
8. Actualizaciones de la publicación NOF Programmer's Guide (guía del programador de NOF) - Verbos adicionales para definir y consultar parámetros del temporizador HPR

Esta nueva función se describe con más detalle a continuación.

1. Soporte de kernel 2.6

CS Linux y CS Linux sobre zSeries ahora soportan instalaciones de servidor en distribuciones de kernel de Linux 2.6 de Red Hat y SuSE. Las distribuciones soportadas para las instalaciones de servidor son RHEL 4 y SLES 9-SP1. Cuando se instala el producto CS Linux en un sistema Intel, el kernel debe ser una distribución Linux de 32 bits. Cuando se instala CS Linux en un Linux sobre un sistema Power (OpenPower o Power5), el kernel debe ser una distribución Linux de 64 bits. El producto CS Linux sobre zSeries se puede instalar en un kernel de Linux de 31 bits o de 64 bits. No obstante, en el futuro, las distribuciones de Linux para la plataforma zSeries solamente serán para el kernel de 64 bits. Las aplicaciones pueden utilizar bibliotecas de 32 bits o de 64 bits.

Para el componente Remote API Client, que ha tenido soporte de distribución de kernel de Linux 2.6, las aplicaciones SNA clientes solamente pueden llamar a bibliotecas de 32 bits en las plataformas Intel. Por lo tanto, si la aplicación cliente se ejecuta en una distribución de Linux x86_64, solamente se podrán utilizar las bibliotecas de 32 bits. Para Linux sobre zSeries y Linux sobre plataformas Power, las aplicaciones SNA clientes pueden utilizar tanto bibliotecas de 32 bits como de 64 bits.

Las distribuciones de kernel de Linux que están soportadas en las diversas plataformas se describen en la tabla siguiente:

Tabla 1. Distribuciones de kernel de Linux soportadas

Plataforma	RHAS 2.1	SLES 8	RHEL 3	SLES 9	RHEL 4	AIX V5
Intel						
Cliente (i686 - 32 bits)	x	x	x	x	x	

Tabla 1. Distribuciones de kernel de Linux soportadas (continuación)

Plataforma	**RHAS 2.1**	**SLES 8**	**RHEL 3**	**SLES 9**	**RHEL 4**	**AIX V5**
Cliente (x86_64), API de 32 bits		x	x	x	x	
Servidor (i686 - 32 bits)	x	x	x	x	x	
OpenPower o Power 5 (kernel de 64 bits, ppc64)						
Cliente				x	x	
Servidor				x	x	
Linux sobre zSeries (s390, 31 bits)						
Cliente		x	x	x	x	
Servidor		x	x	x*	x*	
Linux sobre zSeries (s390x, zSeries, 64 bits)						
Cliente		x	x	x	x	
Servidor		x	x	x	x	
AIX						
Cliente - 32 bits						x
Cliente - 64 bits						x

* indica que el soporte podría no estar disponible ya en los futuros releases de Linux

2. Linux sobre Power

El producto CS Linux (5724-i33) ahora tiene soporte para múltiples plataformas que incluye el servidor CS Linux para Linux en plataformas Power, que corresponden a los servidores OpenPower y a los servidores pSeries Power5. Cuando se instala el servidor, el soporte para estas plataformas requiere RHEL 4 o SLES 9-SP1. Para instalar un servidor CS Linux en un Linux sobre Power, consulte el archivo README.ppc64.xx_yy (donde xx_yy es el entorno local para el sistema).

El cliente de API remota de CS Linux para Linux sobre Power se proporciona con los productos CS Linux y CS Linux para zSeries. El nuevo soporte de cliente se encuentra en el directorio /ibm-commserver-clients/linux-ppc64 del disco de instalación. El cliente de API remota se ejecuta en el ″Espacio de usuario″ en Linux y no es dependiente del kernel. Para instalar el cliente, consulte el archivo /ibm-commserver-clients/linux-ppc64/README.

3. Cliente de API remota AIX

Los servidores Communications Server que se ejecutan en un dominio de servidores que soportan clientes de API remota ahora pueden proporcionar soporte para aplicaciones SNA que se ejecuten en plataformas AIX V5. Cuando se escriban aplicaciones para la plataforma AIX, se deberá utilizar la misma descripción para la compilación y el enlace (linkage) que la que se encuentra en Communications Server para AIX (http://www.ibm.com/software/network/commserver/library/aix).

La conexión entre el cliente AIX y los servidores Linux se realizará sobre TCP/IP. Cualquier aplicación SNA sobre AIX que utilice las interfaces CPI-C, APPC, LUA y alguna NOF para Communications Server para AIX V5 y versiones posteriores, puede utilizar este cliente de API remota de AIX. El cliente de API remota no soporta las interfaces CS/AIX V4.2 más antiguas.

4. Soporte de LU primaria

Las aplicaciones de LUA habitualmente se conectan a grandes sistemas principales como LUs secundarias. Esto significa que la definición para la sesión está controlada por la aplicación del sistema principal, que envía el BIND para iniciar una sesión. Communications Server ahora tiene la posibilidad de descargar los dispositivos dependientes de SNA sobre las interfaces de LAN utilizando la interfaz RUI_INIT_PRIMARY. Utilizando esta interfaz, una aplicación puede conectar sesiones de LU dependientes de descarga sin la necesidad de un gran sistema principal.

Para utilizar las aplicaciones de LU primaria, el nodo debe estar configurado con LU de descarga (o una plantilla PU de descarga) que tenga un nombre de LU de sistema principal de #PRIRUI#. Esto indicará al servidor que las aplicaciones que utilizan RUI_INIT_PRIMARY controlarán esas PU y los recursos asignados a ellas. Las PU solamente se pueden utilizar en puertos LAN. Consulte las actualizaciones (más adelante) de la publicación LUA Programming Guide para la información de interfaz para aplicaciones de programación para utilizar soporte de LU primario.

5. Actualizaciones de clientes de API remota

5.1 Cambio de nombre

El programa Communications Server Remote API Client ahora mostrará el nombre ″IBM Remote API Client″ en lugar del nombre anterior.

5.2 Instalación del cliente de Linux y AIX

Si está realizando la instalación del servidor o cliente de Communications Server en un sistema que ya tenga una versión anterior del código, deberá desinstalar el servidor o cliente en primer lugar. El proceso de desinstalación y de instalación no suprimirá o modificará los archivos de configuración.

5.3 Clientes de Windows

Si está realizando la instalación del cliente de Communications Server en un sistema que ya tenga una versión anterior del código, deberá llevar a cabo los pasos siguientes para retener la configuración actual:

- Detenga el servicio de cliente de Communications Server mediante el mandato **net stop sxclient**. Este mandato se debe escribir tal como aparece aquí (en inglés).
- Cierre el icono del supervisor del cliente de Windows (habitualmente se encuentra en la parte inferior derecha del escritorio).
- Instale el producto sin eliminar la versión anterior.

Si elimina el cliente de Windows anterior que está instalado, la información de configuración se suprimirá de la base de datos del registro de Windows y la información de configuración se deberá entrar de nuevo después de que se instale la versión más reciente.

El componente IBM Remote API Client para Windows ahora incluye una herramienta de diagnóstico para el servicio, **snagetpd.exe**. Esta herramienta crea un archivo compacto, **snapd.exe**, que es autoextraíble. El archivo de determinación de problemas contiene:

- Contenido de registro relevante
- Contenido de directorio de todo los elementos que hay en el directorio de instalación y por debajo de él.
- Información de versión de archivo de todos los archivos binarios instalados
- Ubicaciones de todos los archivos de registro cronológico y de rastreo
- Salida de los mandatos 'ver', 'ipconfig /all', 'route print', 'netstat -an', 'netstat -a'

La herramienta copia todos los archivos de registro cronológico y de rastreo en snapd. Cuando el servicio de Communications Server Linux le indique que proporcione información de determinación de problemas, deberá enviar un archivo snapd.exe. Éste servirá como ayuda en el momento de proporcionar servicio técnico para los problemas informados.

Las correcciones de APAR que se han incluido para el componente Windows Remote API Client son:

- LI70604, LI70605 - La instalación de cancela anormalmente en algunos sistemas Windows XP que no tienen establecido uno de los entornos nacionales por omisión.
- LI70677, LI70678 - La hebra principal de la aplicación WinCPIC no puede emitir más de un ALLOCATE.

6. Actualizaciones de la guía del programador de LUA

La información que sigue corresponde a las adiciones que se han realizado en la publicación LUA Programmer's Guide para describir la interfaz de programa para RUI_INIT_PRIMARY. Deberá consultar al sección para RUI_INIT de la publicación LUA Programmer's Guide para consultar cualquier consideración o característica que no esté completamente descrita en esta sección.

6.1 Diseño y escritura de aplicaciones de LUA: Información de SNA para RUI primaria

Esta sección contiene consideraciones de SNA para la escritura de aplicaciones de RUI primarias de CS Linux para comunicaciones con una LU de descarga.

Esta guía no pretende explicar los conceptos de SNA en detalle. Si necesita información específica sobre flujos de mensajes de SNA, consulte la documentación de la aplicación del sistema principal para el que está diseñando la aplicación de LUA de CS Linux.

6.2 Responsabilidades de la aplicación de RUI primaria

Una aplicación RUI primaria tiene control tanto de sesiones LU-SSCP como PLU-SLU al nivel de Unidad de petición/respuesta (RU), y puede enviar y recibir las RU de SNA en estas sesiones. La sesión PU-SSCP es interna en CS Linux y la aplicación de RUI primaria no puede acceder a ella.

Puesto que la aplicación de RUI primaria trabaja al nivel de RU, tiene un grado de control grande sobre el flujo de datos que va y viene a la LU secundaria. Sin embargo, toma una responsabilidad mayor que una aplicación de LUA corriente para asegurar que los mensajes de SNA que envía son válidos, y que los protocolos de nivel de RU (por ejemplo la delimitación y el encadenamiento) se utilicen de manera correcta. En particular, observe que CS Linux no intenta verificar la validez de las RU enviadas por una aplicación de RUI primaria.

La aplicación de RUI primaria es responsable de:

- Inicializar las LU de descarga utilizando RUI_INIT_PRIMARY, y realizar su terminación utilizando RUI_TERM

- Procesar los mensajes NOTIFY de la LU secundaria como inicio y detención de aplicaciones secundarias
- Procesar INIT-SELF y TERM-SELF para activar y desactivar la sesión PLU-SLU
- Crear, enviar, recibir y analizar mensajes de corriente de datos 3270 en las RU de datos
- Implantar protocolos de nivel de RU (control de petición, delimitación, encadenamiento, dirección)
- Cifrado (si se precisa)
- Compresión (si se precisa).

6.3 Restricciones

CS Linux no soporta los aspectos siguientes para aplicaciones de RUI primaria:

- Las PU de descarga sobre DLUR
- Las LU dependientes definidas dinámicamente (DDDLU)
- El envío de STSN (para restablecer números de secuencia, la aplicación deberá desvincular y volver a vincular (UNBIND y BIND) la sesión).

6.4 Información de configuración: Configuración de enlace de RUI_INIT_PRIMARY

Para una aplicación de RUI primaria en comunicación con una LU de descarga, la única configuración necesaria es la de la LU de descarga (o una plantilla de PU de descarga) que tenga un nombre de LU de sistema principal de #PRIRUI#.

6.5 RUI_VERBS: Descripción del control de verbo RUI_INIT_PRIMARY

El verbo RUI_INIT_PRIMARY establece la sesión SSCP-LU para una aplicación primaria de SNA que se comunica con una LU de descarga. Si la aplicación de RUI actúa como una aplicación secundaria de SNA y se comunica con una LU de sistema principal, debe utilizar RUI_INIT en lugar de RUI_INIT_PRIMARY.

6.5.1 Parámetros proporcionados

La aplicación proporciona los siguientes parámetros (Vea /opt/ibm/sna/include/lua_c.h):

lua_verb
LUA_VERB_RUI

lua_verb_length
La longitud en bytes del registro del verbo de LUA. Establézcala en sizeof(LUA_COMMON).

lua_opcode
LUA_OPCODE_RUI_INIT_PRIMARY

lua_correlator
LUA_OPCODE_RUI_INIT_PRIMARY

lua_luname
El nombre en ASCII de la LU para la que desea iniciar la sesión. Éste debe coincidir con el nombre de una LU de descarga configurada para el uso con una Pasarela SNA o una LU creada implícitamente desde una plantilla de LU de descarga que tenga un nombre de LU de sistema principal de #PRIRUI#.

Este parámetro debe tener una longitud de ocho bytes; rellene el lado derecho con espacios en blanco, 0x20, si el nombre es menor de ocho caracteres.

lua_max_length
La longitud de un almacenamiento intermedio proporcionado para recibir una copia de la RU de ACTLU(+RSP) recibida desde la PU de descarga. Si la aplicación no necesita recibir esta información, puede especificar un puntero nulo en el parámetro lua_data_ptr, en cuyo caso no es necesario proporcionar un almacenamiento intermedio de datos.

lua_data_ptr
Un puntero al almacenamiento intermedio proporcionado para recibir una copia de la RU de ACTLU(+RSP) recibida desde la PU de descarga. Si la aplicación no necesita recibir esta información, puede especificar un puntero nulo, y no se devolverá la información.

lua_post_handle
Un puntero a una rutina de devolución de llamada. Si el verbo finaliza de manera asíncrona, LUA llamará a esta rutina para indicar la finalización del verbo. Para obtener más información, consulte el apartado Diseño y escritura de aplicaciones de LUA.

lua_encr_decr_option

0 No se utiliza el cifrado a nivel de sesión.

128 Se realiza el cifrado y descifrado.

Nota: El programa de aplicación realiza el cifrado y el descifrado.

Cualquier otro valor producirá el código de retorno LUA_ENCR_DECR_LOAD_ERROR.

6.5.2 Parámetros devueltos

LUA siempre devuelve el parámetro siguiente:

lua_flag2.async
Este distintivo se establece en 1 si el verbo ha finalizado de manera asíncrona, o bien en 0 si el verbo ha finalizado de manera síncrona. RUI_INIT_PRIMARY también finalizará de manera asíncrona salvo que devuelva un error tal como LUA_PARAMETER_CHECK.

Otros parámetros devueltos dependen de si el verbo finaliza de manera satisfactoria; vea las secciones que siguen.

Ejecución satisfactoria

Si el verbo se ejecuta de manera satisfactoria, LUA devuelve los parámetros siguientes.

lua_prim_rc
LUA_OK

lua_sid
Un ID de sesión para al nueva sesión. Este ID lo pueden utilizar los verbos subsiguientes para identificar esta sesión.

lua_data_length
La longitud del RU de ACTLU(+RSP) recibida de la PU de descarga. LUA coloca los datos en el almacenamiento intermedio especificado por lua_data_ptr.

Ejecución no satisfactoria

Si el verbo no finaliza de manera satisfactoria, LUA devuelve un código de retorno primario para indicar el tipo de error y un código de retorno secundario para proporcionar detalles específicos en relación a la razón para la ejecución no satisfactoria.

Por favor, consulte la sección de la publicación LUA Programmer's Guide que describe la ejecución no satisfactoria para RUI_INIT, para ver la lista de posibles códigos de retorno. Además de esto, se pueden devolver las indicaciones siguientes para errores específicos de RUI_INIT_PRIMARY:

lua_prim_rc
LUA_STATE_CHECK

lua_sec_rc
LUA_DUPLICATE_RUI_INIT_PRIMARY

Se está procesando actualmente un verbo RUI_INIT_PRIMARY para esta sesión.

6.5.3 Interacción con otros verbos

El verbo RUI_INIT_PRIMARY debe ser el primer verbo de LUA emitido para la sesión.

Mientras que este verbo no haya finalizado de manera satisfactoria, el único otro verbo de LUA que se puede emitir para esta sesión es RUI_TERM, que terminará un RUI_INIT_PRIMARY pendiente.

El resto de los verbos emitidos en esta sesión deben identificar la sesión utilizando uno de los parámetros siguientes de este verbo:

- El ID de sesión, devuelto a la aplicación en el parámetro lua_sid
- El nombre de LU, proporcionado por la aplicación en el parámetro lua_luname

6.5.4 Uso y restricciones

El verbo RUI_INIT_PRIMARY finaliza después de una respuesta positiva de ACTLU recibida de la LU de descarga. Si es necesario, el verbo espera indefinidamente. Si la aplicación necesita comprobar el contenido de esta respuesta positiva de ACTLU, puede hacerlo proporcionando un almacenamiento intermedio de datos en RUI_INIT_PRIMARY (utilizando los parámetros lua_max_length y lua_data_ptr) en los que CS Linux devuelve el contenido del mensaje recibido.

Una vez que el verbo RUI_INIT_PRIMARY haya finalizado satisfactoriamente, esta sesión utilizará la LU para la que se inició la sesión. Ninguna otra sesión de LUA (desde esta o cualquier otra aplicación) podrá utilizar la LU mientras el verbo RUI_TERM esté emitido, o hasta que no se reciba un código de retorno primario LUA_SESSION_FAILURE.

Si el verbo RUI_INIT_PRIMARY se devuelve con un código de retorno primario LUA_IN_PROGRESS, entonces el ID de sesión se devolverá en el parámetro lua_sid. Este ID de sesión es el mismo que el devuelto cuando el verbo finaliza satisfactoriamente y se puede utilizar con el verbo RUI_TERM para finalizar un verbo RUI_INIT_PRIMARY pendiente.

7. Actualizaciones de la guía de administración

Existen 2 nuevos mandatos para la herramienta de administración snaadmin. Estos mandatos son:

define_rtp_tuning
definir nuevos temporizadores para el ajuste de conectividad de sesión HPR.

query_rtp_tuning
consultar temporizadores de conmutación de vía de acceso de HPR

Para utilizar estos mandatos, emita el mandato de ayuda ″snaadmin -d -h define_rtp_tuning″ o bien ″snaadmin -d -h query_rtp_tuning″ para ver la sintaxis del mandato. También puede consultar las actualizaciones de la publicación NOF Programmer's Guide para obtener información más específica.

8. Actualizaciones de la guía del programador de NOF

8.1 DEFINE_RTP_TUNING

DEFINE_RTP_TUNING especifica los parámetros a utilizar cuando se configuran conexiones RTP. Después de emitir este verbo, los parámetros que especifique se utilizarán para todas las conexiones RTP futuras mientras no los modifique emitiendo un nuevo verbo DEFINE_RTP_TUNING.

8.1.1 Parámetros proporcionados

La aplicación proporciona los siguientes parámetros (Vea la estructura define_rtp_tuning en /opt/ibm/sna/include/nof_c.h):

opcode
AP_DEFINE_RTP_TUNING

path_switch_attempts
Número de intentos de conmutación de vía de acceso a establecer en nuevas conexiones RTP. Especifique un valor dentro del rango de 1 a 255. Si especifica 0 (cero), CS Linux utilizará el valor por omisión de 6.

short_req_retry_limit
Número de veces que se envía una Petición de estado antes de que CS Linux determine que una conexión RTP se desconecta y se inicia el proceso de Conmutación de vía de acceso. Especifique un valor dentro del rango de 1 a 255. Si especifica 0 (cero), CS Linux utilizará el valor por omisión de 6.

path_switch_times
Duración del tiempo en segundos durante el cual CS Linux intenta conmutar de vía de acceso una conexión RTP desconectada. Este parámetro se especifica como cuatro límites de tiempo separados para cada una de las prioridades de transmisión válidas, en el orden: AP_LOW, AP_MEDIUM , AP_HIGH y AP_NETWORK. Cada una de ellas debe estar dentro del rango de 1 a 65535. El valor que se especifique para cada prioridad de transmisión no debe exceder del valor para cualquier prioridad de transmisión más baja.

Si especifica 0 (cero) para cualquiera de estos valores, CS Linux utilizará el correspondiente valor por omisión, tal como sigue:

- 480 segundos (8 minutos) para AP_LOW
- 240 segundos (4 minutos) para AP_MEDIUM
- 120 segundos (2 minutos) para AP_HIGH
- 60 segundos (1 minuto) para AP_NETWORK

8.1.2 Parámetros devueltos

Ejecución satisfactoria

Si el verbo se ejecuta de manera satisfactoria, CS Linux devuelve los parámetros siguientes:

primary_rc
AP_OK

Ejecución no satisfactoria

Si el verbo no se ejecuta debido a un error de parámetro, CS Linux devuelve los parámetros siguientes:

primary_rc
AP_PARAMETER_CHECK

secondary_rc
Los valores posibles son:

AP_INVALID_PATH_SWITCH_TIMES El parámetro path_switch_times no era válido; por ejemplo, puede haber especificado un valor para una prioridad de transmisión que exceda del valor especificado para una prioridad de transmisión más baja.

Los Códigos de retorno comunes listan más códigos de retorno secundarios asociados con AP_PARAMETER_CHECK, que son comunes a todos los verbos NOF.

8.2 QUERY_RTP_TUNING

QUERY_RTP_TUNING devuelve información sobre los parámetros que se utilizarán para las conexiones RTP futuras. Esta información se ha configurado previamente utilizando DEFINE_RTP_TUNING.

8.2.1 Parámetros proporcionados

La aplicación proporciona los siguientes parámetros (Vea la estructura query_rtp_tuning en /opt/ibm/sna/include/nof_c.h):

opcode
AP_QUERY_RTP_TUNING

8.2.2 Parámetros devueltos

Ejecución satisfactoria

Si el verbo se ejecuta satisfactoriamente, CS Linux devuelve los parámetros siguientes:

primary_rc
AP_OK

path_switch_attempts
Número de intentos de conmutación de vía de acceso a establecer en nuevas conexiones RTP.

short_req_retry_limit
Número de veces que se envía una Petición de estado antes de que CS Linux determine que se desconecta una conexión RTP y se inicia el proceso de Conmutación de vía de acceso.

path_switch_times
Duración del tiempo en segundos para durante el que CS Linux intenta conmutar de vía de acceso una conexión RTP desconectada. Este parámetro se especifica como cuatro límites de tiempo separados para cada una de las prioridades de transmisión válidas, en el orden: AP_LOW, AP_MEDIUM , AP_HIGH y AP_NETWORK.

Otras condiciones

Los Códigos de retorno comunes listan más combinaciones de códigos de retorno primarios y secundarios que son comunes a todos los verbos NOF.