IBM Communications Server for Linux V6.2.1 での新機能

この資料は、Communications Server for Linux (製品番号 5724-i33、CS Linux) および Communications Server for Linux on zSeries (製品番号 5724-i34、zSeries 上の CS Linux) のバージョン 6.2.1.0 に関するメンテナンス・リリース変更を説明しています。説明は、Communications Server for Linux と Communications Server for Linux on zSeries の両製品に適用されます。

この Communications Server for Linux V6.2.1.0 に関するメンテナンス・リリースで提供されている変更には、以下のものが含まれています。

1. 2.6 カーネル・サポート - 更新された Linux オペレーティング・システム・サポート
2. Power 上の Linux - 追加プラットフォーム・サポートおよびパッケージ化
3. AIX Remote API Client - 追加プラットフォーム・パッケージ化
4. Primary LU サポート - Primary LU 0 フローをサポートする LUA プログラミング・インターフェース
5. Remote API Client 更新 - クライアントに提供される更新
6. LUA プログラマーズ・ガイドの更新 - 追加 Primary LU インターフェース
7. 管理ガイドの更新 - HPR タイマー・パラメーターを定義し照会する追加コマンド
8. NOF プログラマーズ・ガイドの更新 - HPR タイマー・パラメーターを定義し照会する追加 verb

この新機能については、以下でより詳細に説明します。

1. 2.6 カーネル・サポート

CS Linux および zSeries 上の CS Linux は、Red Hat および SuSE からの 2.6 Linux カーネル・ディストリビューション上でのサーバー・インストールをサポートするようになりました。サーバー・インストール用にサポートされるディストリビューションは、RHEL 4 および SLES 9-SP1 です。 CS Linux 製品を Intel システムにインストールする場合には、カーネルは 32 ビット Linux ディストリビューションでなければなりません。 CS Linux を Power (OpenPower または Power5) システム上の Linux にインストールする場合には、カーネルは 64 ビット Linux ディストリビューションでなければなりません。 zSeries 製品上の CS Linux は、31 ビットまたは 64 ビットの Linux カーネルにインストールできます。ただし、将来、zSeries プラットフォーム用の Linux ディストリビューションは、64 ビット・カーネル専用になります。アプリケーションは 32 ビットまたは 64 ビットのライブラリーを使用することができます。

2.6 Linux カーネル・ディストリビューション・サポートを備えていた Remote API Client の場合、クライアント SNA アプリケーションは、Intel プラットフォーム上の 32 ビット・ライブラリーだけを呼び出すことができます。したがって、クライアント・アプリケーションが x86_64 Linux ディストリビューション上で実行している場合には、32 ビット・ライブラリーだけを使用することができます。 zSeries 上の Linux および Power プラットフォーム上の Linux の場合、クライアント SNA アプリケーションは 32 ビットまたは 64 ビットのライブラリーを使用することができます。

さまざまなプラットフォーム上でサポートされる Linux カーネル・ディストリビューションが以下の表でマップされています。

表 1. サポートされる Linux カーネル・ディストリビューション

プラットフォーム	RHAS 2.1	SLES 8	RHEL 3	SLES 9	RHEL 4	AIX V5
Intel						
Client (i686 - 32 ビット)	x	x	x	x	x	
Client (x86_64)、32 ビット API		x	x	x	x	
Server (i686 - 32 ビット)	x	x	x	x	x	
OpenPower または Power 5 (64 ビット・カーネル、ppc64)						
Client				x	x	
Server				x	x	
zSeries 上の Linux (s390、31 ビット)						
Client		x	x	x	x	
Server		x	x	x*	x*	
zSeries (s390x、zSeries、 64 ビット) 上の Linux						
Client		x	x	x	x	
Server		x	x	x	x	
AIX						
Client - 32 ビット						x
Client - 64 ビット						x

* 将来の Linux のリリースでは使用すべきでなくなる可能性のあるサポートを示します。

2. POWER 上の Linux

CS Linux 製品 (5724-i33) は、OpenPower サーバーおよび Power5 pSeries サーバーである Power プラットフォーム上の Linux 用 CS Linux サーバーを含むマルチプラットフォーム・サポートを備えるようになりました。サーバーをインストールするときに、これらのプラットフォームに対するサポートには、RHEL 4 かまたは SLES 9-SP1 が必要です。 Power 上の Linux に CS Linux サーバーをインストールするには、README.ppc64.xx_yy (ここで xx_yy はシステムのロケール) を参照してください。

Power 上の Linux 用の CS Linux Remote API クライアントは、CS Linux および zSeries 製品用 CS Linux とともに配送されます。新規のクライアント・サポートは、インストール・ディスクの /ibm-commserver-clients/linux-ppc64 ディレクトリーにあります。 Remote API クライアントは、Linux 上の「ユーザー・スペース」で実行し、カーネル依存ではありません。クライアントをインストールするには、/ibm-commserver-clients/linux-ppc64/README を参照してください。

3. AIX Remote API Client

Remote API クライアントをサポートするサーバーのドメインで実行している複数の Communications Server は、AIX V5 プラットフォームで実行している SNA アプリケーションに対するサポートを提供できるようになりました。 AIX プラットフォーム用のアプリケーションを作成するときには、AIX 用 Communications Server (http://www.ibm.com/software/network/commserver/library/aix) にあるコンパイルとリンクに関する説明と同じ説明を使用する必要があります。

AIX クライアントと Linux サーバーの間の接続は、TCP/IP を使用して行われます。 AIX V5 以降のための Communications Server 用の CPI-C、APPC、LUA およびいくつかの NOF インターフェースを使用する AIX 上の 任意の SNA アプリケーションは、この AIX Remote API Client を使用することができます。古い CS/AIX V4.2 インターフェースは、Remote API Client ではサポートされません。

4. Primary LU サポート

LUA アプリケーションは、通常、セカンダリー LU としてホスト・メインフレームに接続します。これは、セッションの定義が、BIND を送信してセッションを開始するホスト・アプリケーションによって制御されることを意味します。 Communications Server は、RUI_INIT_PRIMARY インターフェースを使用して、LAN インターフェースを介して、ダウンストリーム SNA 依存装置に対する Primary LU としての役割を果たす能力をもつことができるようになりました。アプリケーションは、このインターフェースを使用して、ホスト・メインフレームを必要とせずに、ダウンストリーム依存 LU セッションを接続することができます。

Primary LU アプリケーションを使用するには、ノードはホスト LU 名 #PRIRUI# のダウンストリーム LU (または ダウンストリーム PU テンプレート) を使用して構成される必要があります。これは、RUI_INIT_PRIMARY を使用するアプリケーションが、それらに割り当てられる PU および LU のリソースを制御することをサーバーに示します。 PU は LAN ポートでのみ使用できます。プログラミング・アプリケーションが Primary LU サポートを使用するためのインターフェース情報については、『LUA プログラミング・ガイド 更新』(以下に示す) を参照してください。

5. Remote API Clients の更新

5.1 名前の変更

Communications Server Remote API Client は、以前の名前の代わりに「IBM Remote API Client」という名前で示されるようになりました。

5.2 Linux および AIX Client のインストール

Communications Server のサーバーまたはクライアントを、以前のバージョンのコードをすでにもっているシステムにインストールする場合には、まずそのサーバーまたはクライアントをアンインストールする必要があります。構成ファイルは、アンインストールおよびインストールのプロセスによって削除または変更されることはありません。

5.3 Windows クライアント

Communications Server のクライアントを、以前のバージョンのコードをすでにもっているシステムにインストールする場合には、以下の手順を行って、現在の構成を保存する必要があります。

- **net stop sxclient** を出して Communications Server クライアント・サービスを停止します。このコマンドは、示されているとおりに (変換せずに) 入力する必要があります。
- Windows クライアント・モニター・アイコン (通常、デスクトップの下部の右の部分にある) を閉じます。
- 前のバージョンを除去せずに製品をインストールします。

インストール済みの前の Windows クライアントを除去する場合には、構成情報は Windows レジストリー・データベースから削除されるので、新しいバージョンがインストールされた後で構成情報を再度入力する必要があります。

現在、IBM Remote API Client for Windows には、サービス用の診断ツール **snagetpd.exe** が組み込まれています。このツールは、自己解凍型の圧縮ファイル **snapd.exe** を作成します。問題判別ファイルには、以下のものが含まれています。

- 関連するレジストリーの内容
- インストール・ディレクトリー内の、およびそれより下にあるすべてのものからなるディレクトリーの内容
- すべてのインストール済みバイナリーのファイル・バージョン情報
- すべてのログおよびトレース・ファイルの存在場所
- 「ver」、「ipconfig /all」、「route print」、「netstat -an」、「netstat -a」の各コマンドの出力

このツールは、すべてのログおよびトレースのファイルを snapd にコピーします。 Communications Server Linux サービスから問題判別情報を提供するように指示された場合には、snapd.exe ファイルを送信する必要があります。それは報告された問題用のサービスを提供するのに役立ちます。

Windows Remote API Client のために組み込まれている APAR フィックスは以下のとおりです。

- LI70604、LI70605 - デフォルトのロケール・セットの内の 1 つをもっていない、いくつかの Windows XP システムではインストールは異常終了します。
- LI70677、LI70678 - WinCPIC アプリケーションのメイン・スレッドは、複数の ALLOCATE を出すことはできません。

6. LUA プログラマーズ・ガイドの更新

次の情報は、RUI_INIT_PRIMARY 用のプログラム・インターフェースを説明するための、LUA プログラマーズ・ガイドへの追加です。このセクションで十分に説明できない考慮事項または機能については、LUA プログラマーズ・ガイドの RUI_INIT に関するセクションを参照してください。

6.1 LUA アプリケーションの設計および作成: RUI Primary に関する SNA 情報

このセクションには、ダウンストリーム LU との通信のための CS Linux RUI Primary アプリケーションの作成に関する SNA の考慮事項が記載されています。

このガイドは、SNA の概念を詳細に説明しようとはしていません。 SNA メッセージ・フローに関する特定の情報が必要である場合は、ユーザーの CS Linux LUA アプリケーションを設計する対象のホスト・アプリケーションの資料を参照してください。

6.2 Primary RUI アプリケーションの責任

Primary RUI アプリケーションは、要求/応答単位 (RU) レベルでの LU-SSCP セッションおよび PLU-SLU セッションの両方を制御し、それらのセッションで SNA RU の送受信を行うことができます。 PU-SSCP セッションは CS Linux にとって内部的なものであり、Primary RUI アプリケーションはそれにアクセスすることはできません。

Primary RUI アプリケーションは RU レベルで作動するので、このアプリケーションはセカンダリー LU との間のデータ・フローに対して大幅な制御を行います。しかし、そのアプリケーションは、それが送信する SNA メッセージが有効であること、および RU レベル・プロトコル (たとえばブラケット化およびチェーニング) が正しく使用されていることを確かめるために通常の LUA アプリケーションよりも大きな責任をとります。とくに、CS Linux は、Primary RUI アプリケーションによって送信される RU の妥当性の検証を行おうとはしないことに注意してください。

Primary RUI アプリケーションは、以下のことに対して責任をもちます。

- RUI_INIT_PRIMARY の使用によるダウンストリーム LU の初期化、および RUI_TERM の使用によるそれらの終了。
- セカンダリー・アプリケーションの開始時および停止時の、セカンダリー LU からの NOTIFY メッセージの処理
- PLU-SLU セッションの活動化と非活動化のための INIT-SELF および TERM-SELF の処理
- データ RU における 3270 データ・ストリーム・メッセージの作成、送信、受信、および構文解析
- RU レベル・プロトコルのインプリメント (要求制御、ブラケット化、チェーニング、ダイレクション)
- 暗号化 (必要な場合)
- 圧縮 (必要な場合)

6.3 制約事項

CS Linux は、Primary RUI アプリケーションに関して以下のことはサポートしません。

- DLUR を介するダウンストリーム PU
- 動的に定義される従属 LU (DDDLU)
- STSN の送信 (シーケンス番号をリセットするために、アプリケーションはセッションを UNBIND および 再 BIND する必要があります)。

6.4 構成情報: RUI_INIT_PRIMARY Link 構成

ダウンストリーム LU と通信している Primary RUI アプリケーションの場合、必要な構成は、ホスト LU 名 #PRIRUI# のダウンストリーム LU (またはダウンストリーム PU テンプレート) だけです。

6.5 RUI_VERBS: RUI_INIT_PRIMARY verb 制御の説明

RUI_INIT_PRIMARY verb は、ダウンストリーム LU と通信している SNA Primary アプリケーション用の SSCP-LU セッションを確立します。 RUI アプリケーションが SNA セカンダリーとしての役割を果たし、しかもホスト LU と通信する場合には、そのアプリケーションは、RUI_INIT_PRIMARY ではなく RUI_INIT を使用する必要があります。

6.5.1 提供されるパラメーター

アプリケーションは以下のパラメーターを提供します (/opt/ibm/sna/include/lua_c.h を参照してください)。

lua_verb
LUA_VERB_RUI

lua_verb_length
LUA verb レコードのバイト単位の長さ。これを sizeof(LUA_COMMON) に設定します。

lua_opcode
LUA_OPCODE_RUI_INIT_PRIMARY

lua_correlator
LUA_OPCODE_RUI_INIT_PRIMARY

lua_luname
セッションを開始させたい LU の ASCII による名前。この名前は、SNA Gateway で使用するために構成されたダウンストリーム LU の名前、またはホスト LU 名 #PRIRUI# のダウンストリーム LU テンプレートから暗黙に作成される LU の名前と一致する必要があります。

このパラメーターは、長さが 8 バイトでなければなりません。名前が 8 文字より短い場合には、右にスペース 0x20 を埋め込みます。

lua_max_length
ダウンストリーム PU から受信した ACTLU(+RSP) RU のコピーを受け取るために提供されるバッファーの長さ。アプリケーションがこの情報を受け取る必要がない場合は、アプリケーションは lua_data_ptr パラメーターにヌル・ポインターを指定することができます。その場合には、アプリケーションはデータ・バッファーを提供する必要はありません。

lua_data_ptr
ダウンストリーム PU から受信した ACTLU(+RSP) RU のコピーを受け取るために提供されるバッファーを指すポインター。アプリケーションがこの情報を受け取る必要がない場合は、アプリケーションはヌル・ポインターを指定することができます。その情報は戻されません。

lua_post_handle
コールバック・ルーチンを指すポインター。 verb が非同期的に完了した場合は、LUA は、このルーチンを呼び出して verb の完了を示します。詳細については、「LUA アプリケーションの設計と作成」を参照してください。

lua_encr_decr_option

0 セッション・レベル暗号化は使用されません。

128 暗号化および暗号化解除が実行されます。

注: 暗号化および暗号化解除がアプリケーション・プログラムによって実行されます。

その他の任意の値が用いられると、結果として戻りコード LUA_ENCR_DECR_LOAD_ERROR が戻されます。

6.5.2 戻されるパラメーター

LUA は、常に次のパラメーターを戻します。

lua_flag2.async
このフラグは、verb が非同期的に完了すると 1 に設定され、verb が同期的に完了すると 0 に設定されます。 RUI_INIT_PRIMARY は、LUA_PARAMETER_CHECK といったエラーを戻さない限り、常に非同期的に完了します。

その他の戻されたパラメーターは、verb が正常に完了するかどうかによって異なります。以下のセクションを参照してください。

正常な実行

verb が正常に実行されると、LUA は以下のパラメーターを戻します。

lua_prim_rc
LUA_OK

lua_sid
新規セッションのセッション ID。これは、これ以降の verb でこのセッションを識別するために使用されます。

lua_data_length
ダウンストリーム PU から受信された ACTLU(+RSP) RU の長さ。 LUA は lua_data_ptr によって指定されたバッファーにデータを入れます。

正常に行われなかった実行

verb が正常に完了しなかった場合には、LUA は 1 次戻りコードを戻してエラーのタイプを示し、2 次戻りコードを戻して正常に行われなかった実行の理由に関する特定の詳細を提供します。

LUA プログラマーズ・ガイド の「正常に行われなかった RUI_INIT の実行」を説明するセクションを参照して、起こりうる戻りコードのリストを参照してください。これらに加えて、RUI_INIT_PRIMARY 固有のエラーに対して以下に挙げる指示も戻されることがあります。

lua_prim_rc
LUA_STATE_CHECK

lua_sec_rc
LUA_DUPLICATE_RUI_INIT_PRIMARY

RUI_INIT_PRIMARY verb が現在このセッションについて処理されています。

6.5.3 他の verb との対話

RUI_INIT_PRIMARY verb が、このセッションについて出される最初の LUA verb でなければなりません。

この verb が正常に完了するまでは、このセッションで出すことのできる他の LUA verb は、保留中の RUI_INIT_PRIMARY を終了させる RUI_TERM だけです。

このセッションで出されるその他のすべての verb は、この verb からの以下のパラメーターのうちの 1 つを使用してセッションを識別する必要があります。

- lua_sid パラメーターでアプリケーションに戻される セッション ID
- lua_luname パラメーターでアプリケーションによって提供される LU 名

6.5.4 使用法と制約事項

RUI_INIT_PRIMARY verb は、ダウンストリーム LU から ACTLU 肯定応答が受信された後で完了します。必要な場合、この verb はいつまでも待ちます。アプリケーションがこの ACTLU 肯定応答の内容を検査する必要がある場合には、アプリケーションは、(lua_max_length および lua_data_ptr のパラメーターを使用して) CS Linux が受信したメッセージの内容を入れて戻す RUI_INIT_PRIMARY にデータ・バッファーを提供することによってそれを行うことができます。

いったん RUI_INIT_PRIMARY verb が正常に完了すると、このセッションは、セッションの開始の対象となった LU を使用します。それ以外の LUA セッション (このアプリケーション、またはそれ以外の任意のアプリケーションからの) は、RUI_TERM verb が出されるまで、あるいは LUA_SESSION_FAILURE 1 次戻りコードが受け取られるまでは、その LU を使用することはできません。

RUI_INIT_PRIMARY verb が LUA_IN_PROGRESS 1 次戻りコードを戻す場合には、セッション ID が lua_sid パラメーターによって戻されます。このセッション ID は verb が正常に完了したときに戻されるものと同じであり、RUI_TERM verb が未解決の RUI_INIT_PRIMARY を終了させるのに使用することができます。

7. 管理ガイドの更新

snaadmin 管理ツール用の 2 つの新規コマンドがあります。これらのコマンドは次のとおりです。

define_rtp_tuning
HPR セッション接続性を調整するための新規タイマーを定義します。

query_rtp_tuning
HPR パス・スイッチ・タイマーを照会します。

これらのコマンドを使用するには、ヘルプ・コマンド「snaadmin -d -h define_rtp_tuning」または「snaadmin -d -h query_rtp_tuning」を出してコマンドの構文を参照してください。さらに詳細な情報については、NOF プログラマーズ・ガイドの更新を参照することもできます。

8. NOF プログラマーズ・ガイドの更新

8.1 DEFINE_RTP_TUNING

DEFINE_RTP_TUNING は、RTP 接続をセットアップするときに使用されるパラメーターを指定します。いったんこの verb を出せば、ユーザーが指定したパラメーターは、新しい DEFINE_RTP_TUNING を出してそれらを変更するまでは、将来のすべての RTP 接続で使用されることになります。

8.1.1 提供されるパラメーター

アプリケーションは以下のパラメーターを提供します (define_rtp_tuning structure in /opt/ibm/sna/include/nof_c.h を参照してください)。

opcode
AP_DEFINE_RTP_TUNING

path_switch_attempts
新規の RTP 接続で設定を行うパス・スイッチの試行の数。 1 から 255 の範囲の値を指定します。 0 (ゼロ) を指定すると、CS Linux はデフォルト値 6 を使用します。

short_req_retry_limit
CS Linux が RTP 接続が切断され Path Switch 処理を開始することを決定する前に、Status Request が送信される回数。 1 から 255 の範囲の値を指定します。 0 (ゼロ) を指定すると、CS Linux はデフォルト値 6 を使用します。

path_switch_times
CS Linux が切断された RTP 接続をパス・スイッチする試行の時間の秒単位の長さ。このパラメーターは、AP_LOW、AP_MEDIUM、AP_HIGH、および AP_NETWORK という順序の有効な伝送優先順位

のそれぞれに対する 4 つの別々の制限時間として指定されます。これらは、それぞれ 1 から 65535 の範囲になければなりません。各伝送優先順位に指定する値は、どの、より低い伝送優先順位に対する値も超えてはなりません。

これらの値のいずれかに 0 (ゼロ) を指定すると、CS Linux は以下のように対応するデフォルト値を使用します。

- AP_LOW に対して 480 秒 (8 分)
- AP_MEDIUM に対して 240 秒 (4 分)
- AP_HIGH に対して 120 秒 (2 分)
- AP_NETWORK に対して 60 秒 (1 分)

8.1.2 戻されるパラメーター

正常な実行

verb が正常に実行されると、CS Linux は以下のパラメーターを戻します。

primary_rc
AP_OK

正常に行われなかった実行

パラメーター・エラーのために verb が実行されなかった場合には、CS Linux は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
可能な値は以下のとおりです。

AP_INVALID_PATH_SWITCH_TIMES path_switch_times パラメーターが無効でした。たとえば、より低い伝送優先順位に対して指定された値を超える値を伝送優先順位のうちいずれか 1 つに対して指定した可能性があります。

Common Return Codes (共通戻りコード) は、すべての NOF verb に共通な AP_PARAMETER_CHECK に関連する他の 2 次戻りコードをリストします。

8.2 QUERY_RTP_TUNING

QUERY_RTP_TUNING は、将来の RTP 接続に使用されることになるパラメーターに関する情報を戻します。この情報は、以前は DEFINE_RTP_TUNING を使用してセットアップされていました。

8.2.1 提供されるパラメーター

アプリケーションは以下のパラメーターを提供します (query_rtp_tuning structure in /opt/ibm/sna/include/nof_c.h を参照してください)。

opcode
AP_QUERY_RTP_TUNING

8.2.2 戻されるパラメーター

正常な実行

verb が正常に実行されると、CS Linux は以下のパラメーターを戻します。

primary_rc
AP_OK

path_switch_attempts
新規の RTP 接続で設定を行うパス・スイッチの試行の数。

short_req_retry_limit
CS Linux が RTP 接続が切断され Path Switch 処理を開始することを決定する前に、Status Request が送信される回数。

path_switch_times
CS Linux が切断された RTP 接続をパス・スイッチする試行の時間の秒単位の長さ。このパラメーターは、AP_LOW、AP_MEDIUM、AP_HIGH、および AP_NETWORK という順序の有効な伝送優先順位のそれぞれに対する 4 つの別々の制限時間として指定されます。

他の条件

Common Return Codes (共通戻りコード) は、すべての NOF verb に共通な、1 次戻りコードと 2 次戻りコードの他の組み合わせをリストします。