

IBM Rational Developer for System z
バージョン 9.0.1

ホスト構成リファレンス



IBM Rational Developer for System z
バージョン 9.0.1

ホスト構成リファレンス



お願い

本資料をご使用になる前に、必ず 241 ページの『IBM Rational Developer for System z 資料に関する特記事項』に記載されている情報をお読みください。

本書は、IBM Rational Developer for System z バージョン 9.0.1 (プログラム番号 5724-T07 および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

Order publications by phone or fax. IBM Software Manufacturing Solutions takes publication orders between 8:30 a.m. and 7:00 p.m. eastern standard time (EST). The phone number is (800) 879-2755. The fax number is (800) 445-9269. Faxes should be sent Attn: Publications, 3rd floor.

You can also order publications through your IBM representative or the IBM branch office serving your locality.
NONE:

IBM welcomes your comments. You can send your comments by mail to the following address:

IBM Corporation
Attn: Information Development Department 53NA
Building 501 P.O. Box 12195
Research Triangle Park NC 27709-2195
USA

You can fax your comments to: 1-800-227-5088 (US and Canada)

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

Note to U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： SC14-7290-05
IBM Rational Developer for System z
Version 9.0.1
Host Configuration Reference Guide

発行： 日本アイ・ピー・エム株式会社

担当： トランスレーション・サービス・センター

第1刷 2013.12

© Copyright IBM Corporation 2000, 2013.

目次

図	vii
-------------	-----

表	ix
-------------	----

本書について	xi
------------------	----

本書の対象読者	xi
-------------------	----

変更の要約	xii
-----------------	-----

文書内容の説明	xiv
-------------------	-----

Developer for System z について	xiv
---------------------------------------	-----

セキュリティに関する考慮事項	xiv
--------------------------	-----

TCP/IP に関する考慮事項	xiv
---------------------------	-----

WLM に関する考慮事項	xiv
------------------------	-----

チューニングに関する考慮事項	xiv
--------------------------	-----

パフォーマンスに関する考慮事項	xiv
---------------------------	-----

クライアントへのプッシュの考慮事項	xv
-----------------------------	----

CICSTS に関する考慮事項	xv
---------------------------	----

ユーザー出口に関する考慮事項	xv
--------------------------	----

カスタマイズ、TSO 環境の	xv
--------------------------	----

実行、複数のインスタンスの	xv
-------------------------	----

トラブルシューティング、構成問題の	xv
-----------------------------	----

SSL および X.509 認証のセットアップ	xvi
-----------------------------------	-----

TCP/IP のセットアップ	xvi
--------------------------	-----

IBM Rational Developer for System z ホスト構成リファレンス 1

第 1 章 Developer for System z について 3

コンポーネントの概要	3
----------------------	---

Java アプリケーションとしての RSE	5
---------------------------------	---

タスク所有者	6
------------------	---

接続のフロー	8
------------------	---

統合デバッガー	10
-------------------	----

CARMA	10
-----------------	----

CARMA 構成ファイル	12
------------------------	----

CRASTART	12
--------------------	----

パッチ実行依頼	12
-------------------	----

データ・セット・ロック所有者	13
--------------------------	----

ロックの解放	14
------------------	----

z/OS UNIX ディレクトリー構造	15
-------------------------------	----

非システム管理者の更新特権	17
-------------------------	----

便利なセキュリティ・コマンド	17
--------------------------	----

便利な z/OS UNIX コマンド	18
------------------------------	----

サンプル・セットアップ	18
-----------------------	----

第 2 章 セキュリティーに関する考慮事項 21

認証方式	22
----------------	----

ユーザー ID およびパスワード	22
----------------------------	----

ユーザー ID およびワンタイム・パスワード	22
----------------------------------	----

X.509 証明書	22
---------------------	----

JES ジョブ・モニターの認証	23
---------------------------	----

デバッグ・マネージャーでの認証	23
---------------------------	----

接続セキュリティ	23
--------------------	----

指定したポートのみに限定した外部通信	24
------------------------------	----

SSL または TLS を使用した通信暗号化	24
----------------------------------	----

Port Of Entry 検査	25
----------------------------	----

PassTicket の使用	25
--------------------------	----

監査ログイン	27
------------------	----

監査制御	27
----------------	----

監査処理	27
----------------	----

監査データ	28
-----------------	----

JES セキュリティー	29
-----------------------	----

ジョブに対するアクション - ターゲットの制限	29
-----------------------------------	----

アクション、ジョブに対する - 実行の制限	31
---------------------------------	----

スプール・ファイルへのアクセス	32
---------------------------	----

SSL/TLS 暗号化通信	33
-------------------------	----

統合デバッガーでの暗号化通信	34
--------------------------	----

クライアント認証、X.509 証明書を使用した	35
-----------------------------------	----

認証局 (CA) の妥当性検査	35
---------------------------	----

(オプション) 証明書失効リスト (CRL) に対する	35
---------------------------------------	----

照会	36
--------------	----

使用しているセキュリティ・ソフトウェアによる	36
----------------------------------	----

認証	37
--------------	----

RSE デーモンによる認証	38
-------------------------	----

Port Of Entry (POE) 検査	39
----------------------------------	----

クライアント関数の変更	39
-----------------------	----

OFF.REMOTECOPY.MVS	40
------------------------------	----

クライアントへのプッシュの開発者グループ	41
--------------------------------	----

デバッグ・セキュリティ	42
-----------------------	----

CICSTS セキュリティー	43
--------------------------	----

CRD リポジトリ	43
---------------------	----

CICS トランザクション	43
-------------------------	----

SSL 暗号化通信	43
---------------------	----

各種情報	43
----------------	----

GATE の処分	43
--------------------	----

管理 ACEE	44
-------------------	----

SCLM セキュリティー	44
------------------------	----

Developer for System z 構成ファイル	44
---	----

JES ジョブ・モニター - FEJJCNFG	44
-----------------------------------	----

RSE - rsed.envvars	45
------------------------------	----

RSE - ssl.properties	46
--------------------------------	----

RSE - pushtoclient.properties	47
---	----

セキュリティ定義	47
--------------------	----

要件およびチェックリスト	48
------------------------	----

セキュリティの設定およびクラスをアクティブ	49
---------------------------------	----

にする	49
---------------	----

Developer for System z ユーザーの OMVS セグ	50
--	----

メントを定義する	50
--------------------	----

Developer for System z 開始タスクの定義	51
---	----

セキュアな z/OS UNIX サーバーとして RSE を	51
---	----

定義する	52
----------------	----

RSE の MVS プログラム制御ライブラリーを定義する	53
RSE の PassTicket サポートを定義する	54
RSE のアプリケーション保護の定義	55
JES コマンド・セキュリティを定義する	56
データ・セット・プロファイルを定義する	58
RSE の z/OS UNIX プログラム制御ファイルを定義する	62
セキュリティ設定の検査	63

第 3 章 TCP/IP に関する考慮事項 65

TCP/IP ポート	65
外部通信	66
内部通信	66
TCP/IP ポートの予約	67
CARMA と TCP/IP ポート	67
LDAP の考慮事項	68
TCP/IP のデフォルト動作のオーバーライド	68
遅延 ACK	68
複数スタック (CINET).	68
CARMA とスタックのアフィニティー	69
crastart*.conf	69
CRASUB*	70
Distributed Dynamic VIPA.	70
ポート選択の制限	72
サンプル・セットアップ	74
システム SYS1 - TCP/IP プロファイル	74
システム SYS2 - TCP/IP プロファイル	75

第 4 章 WLM に関する考慮事項 77

ワークロード分類	77
分類規則	78
目標の設定	79
目標の選択に関する考慮事項	80
STC	81
OMVS	82
JES	83
ASCH	84
CICS.	85

第 5 章 チューニングに関する考慮事項 87

リソース使用量	87
概要	88
アドレス・スペースの数	89
プロセスの数	92
スレッドの数	95
一時的なリソース使用量	98
ストレージの使用量	99
Java ヒープ・サイズの限度	99
アドレス・スペース・サイズの限度	100
サイズ見積りのガイドライン	100
ストレージ使用量分析のサンプル	101
z/OS UNIX ファイル・システム・スペースの使用量	105
主要なリソース定義	108
/etc/rdz/rsed.envvars.	108

SYS1.PARMLIB(BPXPRMxx)	109
さまざまなリソース定義	112
サーバー JCL での EXEC カード	112
FEK.#CUST.PARMLIB(FEJCNFG)	112
SYS1.PARMLIB(IEASYSxx).	113
SYS1.PARMLIB(IVTPRMxx)	113
SYS1.PARMLIB(ASCHPMxx)	113
モニター	114
RSE のモニター	114
z/OS UNIX のモニター	115
ネットワークのモニター	117
z/OS UNIX ファイル・システムのモニター	118
サンプル・セットアップ	118
スレッド・プールの数	118
最小限度の特定	119
限度の定義	119
リソース使用量のモニター	121

第 6 章 パフォーマンスに関する考慮事項 123

zFS ファイル・システムの使用	123
STEPLIB の使用の回避	123
改善、システム・ライブラリーへのアクセスの言語環境プログラム (LE) ランタイム・ライブラリー	124
アプリケーション開発	124
向上、セキュリティ検査のパフォーマンスの	125
ワークロード管理	125
固定 Java ヒープ・サイズ	126
Java -Xquickstart オプション	126
JVM 間でのクラス共用	126
クラス共用の有効化	127
キャッシュ・サイズ制限	127
キャッシュ・セキュリティ	127
SYS1.PARMLIB(BPXPRMxx)	128
ディスク・スペース	128
キャッシュ管理ユーティリティー	128

第 7 章 クライアントへのプッシュの考慮事項 131

概要	131
1 次システム	132
クライアントへのプッシュ・メタデータ	133
メタデータのロケーション	133
メタデータのセキュリティ	133
メタデータのスペース使用量	134
クライアント構成の制御	135
クライアント・バージョンの制御	135
複数の開発者グループ	136
アクティベーション	136
グループの連結	137
ワークスペースのバインディング	137
グループ・メタデータのロケーション	138
セットアップ手順	139
LDAP ベースのグループ選択	140
LDAP スキーマ	140

LDAP サーバーの選択	141
LDAP サーバーのロケーション	142
サンプル・セットアップ	142
クライアントへのプッシュ・バックエンドの	
LDAP への追加	143
LDAP グループの初期セットアップ	143
LDAP グループへの開発者の追加	144
pushtoclient.properties	144
rsed.envvars	145
/var/rdz/pushtoclient/*install	145
SAF ベースのグループ選択	145
サンプル・セットアップ	146
セキュリティ定義	147
pushtoclient.properties	147
rsed.envvars	147
/var/rdz/pushtoclient/*install	147
変更の拒否の猶予期間	148
ホスト・ベースのプロジェクト	148
第 8 章 CICSTS に関する考慮事項	151
RESTful と Web サービス	152
主接続領域と非主接続領域	152
CICS リソース・インストール・ロギング	153
Application Deployment Manager セキュリティー	153
CRD リポジトリ・セキュリティ	153
パイプライン・セキュリティ	153
トランザクション・セキュリティ	154
SSL 暗号化通信	155
リソース・セキュリティ	155
管理ユーティリティー	155
管理ユーティリティーのマイグレーションに関する注	159
管理ユーティリティーのメッセージ	160
I CICS トランザクションのデバッグ	163
第 9 章 ユーザー出口に関する考慮事項	165
ユーザー出口の特性	165
ユーザー出口の活動化	165
ユーザー出口ルーチンの作成	165
コンソール・メッセージ	166
可変ユーザー ID を使用した実行	166
z/OS UNIX シェル・スクリプト	166
z/OS UNIX REXX exec	167
使用可能な出口点	168
audit.action	168
logon.action	168
第 10 章 カスタマイズ、TSO 環境の	171
TSO コマンド・サービス	171
アクセス方式	171
TSO/ISPF クライアント・ゲートウェイ・アクセス	
方式の使用	172
ISPF.conf	172
既存の ISPF プロファイルの使用	172
使用、割り振り exec の	173
複数の割り振り exec の使用	174

複数の Developer for System z セットアップで	
の複数の ISPF.conf ファイル	174
第 11 章 実行、複数のインスタンスの	177
同一セットアップ、シスプレックス全体	177
同一のソフトウェア・レベル、異なる構成ファイル	178
自動同期	179
その他のすべての状態	180
第 12 章 トラブルシューティング、構成問題の	183
ログとセットアップの分析に使用、FEKLOGS	184
ログ・ファイル	184
JES ジョブ・モニター・ロギング	186
RSE デーモンおよびスレッド・プールのロギング	186
RSE ユーザー・ロギング	187
SCLM Developer Toolkit のロギング	188
CARMA ロギング	188
fekfivpc IVP テスト・ロギング	189
fekfivpi IVP テスト・ロギング	189
fekfivps IVP テスト・ロギング	189
コード・レビューのロギング	190
コード・カバレッジのロギング	190
ダンプ・ファイル	190
MVS ダンプ	190
Java ダンプ	191
z/OS UNIX ダンプ・ロケーション	192
トレース	193
JES ジョブ・モニターのトレース	193
RSE トレース	193
CARMA トレース	194
エラー・フィードバック・トレース	195
z/OS UNIX 許可ビット	196
SETUID ファイル・システム属性	196
プログラム制御許可	196
APF 許可	198
スティッキー・ビット	199
予約済み TCP/IP ポート	199
アドレス・スペース・サイズ	201
始動 JCL の要件	201
SYS1.PARMLIB(BPXPRMxx) で設定される制限	201
セキュリティ・プロファイル内に保管される制限	201
システム出口によって強制される制限	202
64 ビット・アドレッシングでの制限	202
各種情報	202
エラー・フィードバック B37 スペース異常終了	202
システム限度	203
接続の拒否	203
OutOfMemoryError	203
Host Connect Emulator	204

第 13 章 SSL および X.509 認証のセ ットアップ 205

暗号化方式として SSL または TLS のどちらを使 用するかを決定する	206
秘密鍵と証明書を保管する場所の決定	206
RACF による鍵リングの作成	207
(オプション) 署名付き証明書の使用	208
既存の RSE セットアップのクローン作成	209
共存を可能にするための rsed.envvars の更新	210
SSL を有効にするための ssl.properties の更新	210
新しい RSE デーモンの作成による SSL のアクテ ィブ化	211
接続のテスト	211
(オプション) X.509 クライアント認証サポートの追 加	214
(オプション) gskkyman による鍵データベースの作 成	215
(オプション) keytool による鍵ストアの作成	217

第 14 章 TCP/IP のセットアップ . . . 221

ホスト名依存関係	221
リゾルバーについて	222

構成情報の検索順序について	222
z/OS UNIX 環境で使用される検索順序	223
基本リゾルバー構成ファイル	223
変換テーブル	224
ローカル・ホスト・テーブル	224
このセットアップ情報の Developer for System z へ の適用	225
ホスト・アドレスが正しく解決されない場合	228

参考文献 231

参考資料	231
情報資料	234

用語集 235

IBM Rational Developer for System z 資料に関する特記事項 241

著作権使用許諾	243
商標の帰属表示	244

索引 245



1.	コンポーネントの概要	3
2.	Java アプリケーションとしての RSE	5
3.	タスク所有者	7
4.	接続のフロー	8
I 5.	統合デバッガー	10
6.	CARMA フロー	11
7.	データ・セットのエンキュー判別のフロー	13
8.	z/OS UNIX ディレクトリー構造	15
I 9.	デバッグ・マネージャーの AT-TLS ポリシー	35
10.	TCP/IP ポート	65
11.	update.sh - ファイアウォールによる DDVIPA セットアップのサポート	73
12.	Distributed Dynamic VIPA サンプル	74
13.	WLM 分類	77
14.	アドレス・スペースの最大数	91
15.	クライアントごとのアドレス・スペース数	91
16.	プロセスの最大数	93
17.	STCRSE のプロセス数	94
18.	クライアントごとのプロセス数	94
19.	RSE スレッド・プール・スレッドの最大数	97
20.	JES ジョブ・モニター・スレッドの最大数	97
21.	ログオン数 5 の場合のリソース使用量	102
22.	ログオン数 5 の場合のリソース使用量 (続き)	103
23.	PDS メンバー編集時のリソース使用量	104
24.	z/OS UNIX ファイル・システム・スペースの 使用量	106
25.	サンプル・セットアップのリソース使用量	122
26.	LDAP スキーマ定義の例	141
27.	ADNJSPAU - CICSTS 管理ユーティリティー	157
28.	ADNJSPAU - CICSTS 管理ユーティリティー (2/3)	158
29.	ADNJSPAU - CICSTS 管理ユーティリティー (3/3)	159
30.	RSEDSSL - SSL 用の RSE デーモン・ユーザ ー・ジョブ	211
31.	「ホスト証明書のインポート」ダイアログ	212
32.	「設定」ダイアログ - 「SSL」	213

表

1.	JES ジョブ・モニターのコンソール・コマンド	29	22.	ユーザーごとに必要なリソース使用量	88
2.	LIMIT_COMMANDS コマンドの許可のマトリックス	30	23.	ユーザーごとのリソース使用量	89
3.	拡張 JESSPOOL プロファイル	30	24.	アドレス・スペースの数	89
4.	LIMIT_CONSOLE コンソールの権限マトリックス	31	25.	アドレス・スペースの限度	92
5.	LIMIT_VIEW のブラウズ権限のマトリックス	32	26.	プロセスの数	92
6.	SSL 証明書の保管メカニズム	33	27.	プロセスの限度	95
7.	クライアント関数の変更のための SAF 情報	40	28.	スレッドの数	95
8.	クライアントへのプッシュの SAF 情報	41	29.	スレッドの限度	98
9.	デバッグ機能のための SAF 情報	42	30.	ストレージ使用量に関する参照設定	101
10.	セキュリティ・セットアップの変動要素	48	31.	ログ出力ディレクティブ	107
11.	JES2 ジョブ・モニターのオペレーター・コマンド	56	32.	一時出力ディレクティブ	108
12.	JES3 ジョブ・モニターのオペレーター・コマンド	57	33.	*.enabled に関するクライアントへのプッシュのグループ・サポート・マトリックス	136
13.	WLM エントリー・ポイント・サブシステム	78	34.	reject.*.updates に関するクライアントへのプッシュのグループ・サポート・マトリックス	136
14.	WLM 作業修飾子	79	35.	クライアントへのプッシュ・グループの連結	137
15.	WLM ワークロード	80	36.	クライアントへのプッシュの LDAP 情報	140
16.	WLM ワークロード - STC	81	37.	クライアントへのプッシュの SAF 情報	145
17.	WLM ワークロード - OMVS	82	38.	JAVA_DUMP_TDUMP_PATTERN 変数	191
18.	WLM ワークロード - JES	83	39.	SSL 証明書の保管メカニズム	206
19.	WLM ワークロード - ASCH	84	40.	リゾルバーで使用可能なローカル定義	227
20.	WLM ワークロード - CICS	85	41.	参考資料	231
21.	共通のリソース使用量	88	42.	参照される Web サイト	233
			43.	情報資料	234

本書について

この資料では、IBM® Rational® Developer for System z® 本体と、その他の z/OS® コンポーネントおよび製品 (WLM や CICS® など) の各種構成作業の背景情報について説明しています。

これ以降、本書では以下の名前が使用されています。

- *IBM Rational Developer for System z* は *Developer for System z* と呼ばれます。
- *IBM Rational Developer for System z* 統合デバッガー は統合デバッガーと呼ばれます。
- 共通アクセス・リポジトリ・マネージャー は、*CARMA* と省略されます。
- *Software Configuration and Library Manager Developer Toolkit* は *SCLM Developer Toolkit* と呼ばれ、*SCLMDT* と省略されます。
- *z/OS UNIX* システム・サービス は、*z/OS UNIX* と呼ばれます。
- 顧客情報管理システム (*CICS*) *Transaction Server* は、*CICSTS* と呼ばれ、*CICS* と略されます。

本書は、Developer for System z のホスト構成を説明した文書セットの一部です。これらの文書は、それぞれ特定の読者を対象としています。Developer for System z の構成を行うためにすべての資料に目を通す必要はありません。

- 「*Rational Developer for System z* ホスト構成ガイド」(SC88-5663) では、すべての計画タスク、構成タスクおよびオプション (任意のオプションも含めて) について詳しく説明し、代替方法も記載しています。
- 「*Rational Developer for System z* ホスト構成リファレンス」(SA88-4226) では、Developer for System z 設計について説明し、Developer for System z、z/OS コンポーネント、および Developer for System z 製品に関連したその他の製品 (WLM および CICS など) の各種構成タスクに関する背景情報を提供します。
- 「*Rational Developer for System z* ホスト構成クイック・スタート・ガイド」(GI88-4171) では、Developer for System z の最小限のセットアップについて説明します。
- 「*Rational Developer for System z* ホスト構成ユーティリティー・ガイド」(SA88-4197) では、ホスト構成ユーティリティーについて説明します。このユーティリティーは、Developer for System z の基本的な共通オプションのカスタマイズ・ステップを案内する ISPF パネル・アプリケーションです。

本書の情報は、すべての IBM Rational Developer for System z バージョン 9.0 パッケージに適用されます。

本書の対象読者

本書は、IBM Rational Developer for System z バージョン 9.0.1 を構成およびチューニングするシステム・プログラマーを対象としています。

実際の構成は他の資料に記載されていますが、本書では、チューニング、セキュリティのセットアップなど、各種の関連テーマについて詳しく挙げています。本書を使用するには、z/OS UNIX システム・サービスおよび MVS™ ホスト・システムに精通している必要があります。

変更の要約

ここでは、「*IBM Rational Developer for System z* バージョン 9.0 ホスト構成リファレンス」(SA88-4226-06) (2013 年 12 月更新) での変更点を要約します。

本文または図表に対して技術的な変更または追加が行われている場合には、その個所の左側に縦線を引いて示してあります。

新しい情報:

- タイム・スタンプを持つログ・ファイル名に関する情報が追加されました。184 ページの『ログ・ファイル』を参照してください。
- 新しい監査可能イベントに関する情報が追加されました。監査データを参照してください。

本書には、「*IBM Rational Developer for System z* バージョン 9.0 ホスト構成リファレンス」(SA88-4226-05) に記載されていた情報が含まれています。

新しい情報:

- TCP/IP ポートの使用方法が更新されました。65 ページの『TCP/IP ポート』を参照してください。
- 2 つの RSE デーモンを自動的に同期するサンプルが追加されました。179 ページの『自動同期』を参照してください。
- 新しいログ・ファイルに関する情報が追加されました。184 ページの『ログ・ファイル』を参照してください。

本書には、「*IBM Rational Developer for System z* バージョン 8.5.1 ホスト構成リファレンス」(SA88-4226-03) に記載されていた情報が含まれています。

新しい情報:

- クライアント関数を変更するための、SAF プロファイルに関する情報が追加されました。39 ページの『クライアント関数の変更』を参照してください。
- リソースの使用量が更新されました。87 ページの『第 5 章 チューニングに関する考慮事項』を参照してください。
- スレッド・プールごとの最大ユーザー数のデフォルト値が更新されました。87 ページの『第 5 章 チューニングに関する考慮事項』を参照してください。

本書には、「*IBM Rational Developer for System z* バージョン 8.5 ホスト構成リファレンス」(SA88-4226-02) に記載されていた情報が含まれています。

新しい情報:

- JES ジョブ・モニターのセキュリティ情報が更新されました。21 ページの『第 2 章 セキュリティに関する考慮事項』を参照してください。
- ユーザー出口に関する情報が追加されました。165 ページの『第 9 章 ユーザー出口に関する考慮事項』を参照してください。

本書には、「*IBM Rational Developer for System z* バージョン 8.0.3 ホスト構成リファレンス」(SA88-4226-01)に記載されていた情報が含まれています。

新しい情報:

- z/OS UNIX ディレクトリー構造が更新されました。15 ページの『z/OS UNIX ディレクトリー構造』を参照してください。
- ホスト・ベースのクライアント・コントロールに関する情報が追加されました。131 ページの『第 7 章 クライアントへのプッシュの考慮事項』を参照してください。
- セキュリティー関連のクライアントへのプッシュに関する情報が追加されました。41 ページの『クライアントへのプッシュの開発者グループ』を参照してください。
- 管理 ACEE の使用量を記載しました。44 ページの『管理 ACEE』を参照してください。
- 監査ログ処理の自動化に関する情報が追加されました。27 ページの『監査処理』を参照してください。
- 構成ファイル内のセキュリティーおよび監査関連のディレクティブに関する情報が更新されました。44 ページの『Developer for System z 構成ファイル』を参照してください。
- TCP/IP の情報がさらに追加されました。65 ページの『第 3 章 TCP/IP に関する考慮事項』を参照してください。
- SSL 通信についての認証局情報が更新されました。205 ページの『第 13 章 SSL および X.509 認証のセットアップ』を参照してください。
- リソースの使用量が更新されました。87 ページの『リソース使用量』を参照してください。

本書には、「*IBM Rational Developer for System z* バージョン 8.0.1 ホスト構成リファレンス」(SA88-4226-00)に記載されていた情報が含まれています。

新しい情報:

- 『Developer for System z について』の『CARMA』セクション。10 ページの『CARMA』を参照してください。
- TCP/IP 関連の一般情報。65 ページの『第 3 章 TCP/IP に関する考慮事項』を参照してください。
- B37 スペース異常終了の解決。202 ページの『エラー・フィードバック B37 スペース異常終了』を参照してください。

削除された情報:

- 「*IBM Rational Developer for System z* バージョン 7.6.1 ホスト構成ガイド」(SC88-5663-02)に記載されていた情報は、「*IBM Rational Developer for System z* ホスト構成ガイド」(SC88-5663)と「*IBM Rational Developer for System z* ホスト構成リファレンス」(SA88-4226)の2つに分割されました。
- APPC セットアップに関する情報は、ホワイト・ペーパー「*Using APPC to provide TSO command services*」(SC14-7291)に移されました。
- INETD のセットアップ

文書内容の説明

ここでは、本書に記載されている情報を要約します。

Developer for System z について

Developer for System z ホストは、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。これらのコンポーネントの設計を理解しておく、構成に関して適切な判断を行うことができます。

セキュリティに関する考慮事項

Developer for System z では、メインフレーム以外のワークステーション上にいるユーザーがメインフレームにアクセスできます。このため、接続要求の妥当性検査、ホストとワークステーション間のセキュアな通信の提供、およびアクティビティの許可と監査が、製品構成の重要な側面となります。

TCP/IP に関する考慮事項

Developer for System z では、TCP/IP を使用して、非メインフレーム・ワークステーションのユーザーに、メインフレームからアクセスすることができます。また、各種コンポーネントと他の製品との通信にも TCP/IP が使用されます。

WLM に関する考慮事項

従来の z/OS アプリケーションとは異なり、Developer for System z は、ワークロード・マネージャー (WLM) で容易に識別できる一体構造のアプリケーションではありません。Developer for System z は、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。これらのサービスの一部は異なるアドレス・スペースでアクティブとなるため、WLM 分類も異なることになります。

チューニングに関する考慮事項

RSE (リモート・システム・エクスプローラー) は、Developer for System z の中核です。クライアントからの接続とワークロードを管理するために、RSE は、スレッド・プーリング・アドレス・スペースを制御するデーモン・アドレス・スペースから構成されています。デーモンは接続と管理のためのフォーカル・ポイントとして機能し、スレッド・プールはクライアント・ワークロードを処理します。

このため、RSE は Developer for System z セットアップをチューニングする場合の主要な対象となります。ただし、それぞれが 17 個以上のスレッドを使用する何百人ものユーザー、ある程度の大きさのストレージ、そして場合によっては 1 つ以上のアドレス・スペースを保守するには、Developer for System z と z/OS の両方を適切に構成する必要があります。

パフォーマンスに関する考慮事項

z/OS は高度にカスタマイズ可能なオペレーティング・システムであり、システムの場合によっては小さな) 変更で全体のパフォーマンスに大きな影響を与えることが

できます。この章では、Developer for System z のパフォーマンスを向上させるために行うことができるいくつかの変更を中心に説明します。

クライアントへのプッシュの考慮事項

クライアントへのプッシュ（ホストベースのクライアント制御）では、以下に対する集中管理をサポートしています。

- クライアントの構成ファイル
- クライアントの製品バージョン
- プロジェクト定義

CICSTS に関する考慮事項

この章には、CICS Transaction Server 管理者に有益な情報が記載されています。

ユーザー出口に関する考慮事項

この章は、出口ルーチンの作成による Developer for System z の機能強化についてユーザーを支援します。

カスタマイズ、TSO 環境の

この章は、Developer for System z で TSO 環境に DD ステートメントとデータ・セットを追加することにより、TSO ログオン・プロシージャーを模倣するのに役立ちます。

実行、複数のインスタンスの

同じシステム上で Developer for System z の複数のインスタンスをアクティブにしたい場合があります。例えば、アップグレードをテストするときなどです。しかし、TCP/IP ポートなど、一部のリソースは共用できないため、デフォルトが常に適用可能であるとは限りません。この章の情報を使用して Developer for System z のさまざまなインスタンスの共存を計画してください。その後、この構成ガイドを使用して、それらのインスタンスをカスタマイズすることができます。

トラブルシューティング、構成問題の

この章は、Developer for System z の構成時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのもので、以下のセクションで構成されています。

- ログとセットアップの分析に使用、FEKLOGS
- ログ・ファイル
- ダンプ・ファイル
- トレース
- z/OS UNIX 許可ビット
- 予約済み TCP/IP ポート
- アドレス・スペース・サイズ
- APPC トランザクションおよび TSO コマンド・サービス
- 各種情報

SSL および X.509 認証のセットアップ

この付録は、Secure Socket Layer (SSL) のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。また、この付録には、X.509 証明書で自分自身を認証するユーザーをサポートするためのサンプルのセットアップも記載されています。

TCP/IP のセットアップ

この付録は、TCP/IP のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。

IBM Rational Developer for System z ホスト構成リファレンス

第 1 章 Developer for System z について

Developer for System z ホストは、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。これらのコンポーネントの設計を理解しておく、構成についての正しい決定を行うのに役立ちます。

この章では、以下のトピックについて説明します。

- 『コンポーネントの概要』
- 5 ページの『Java アプリケーションとしての RSE』
- 6 ページの『タスク所有者』
- 8 ページの『接続のフロー』
- 10 ページの『統合デバッガー』
- 10 ページの『CARMA』
- 13 ページの『データ・セット・ロック所有者』
- 15 ページの『z/OS UNIX ディレクトリ構造』

コンポーネントの概要

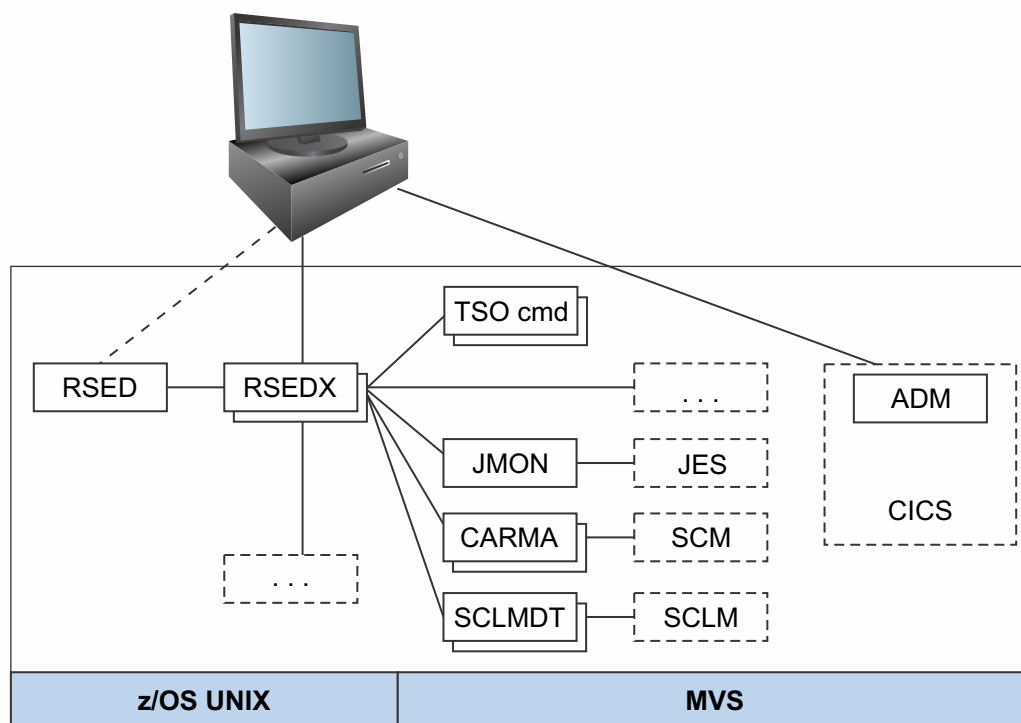


図 1. コンポーネントの概要

3 ページの図 1 は、ホスト・システムにおける Developer for System z のレイアウトの一般的な概要です。

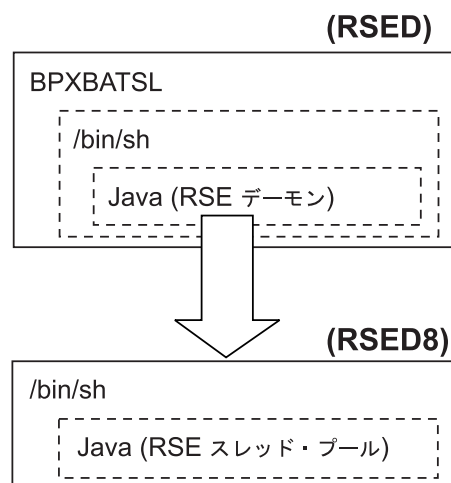
- リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続したり、特定のサービス用に他のサーバーを始動するなどの、コア・サービスを提供します。RSE は、次の 2 つの論理エンティティから構成されます。
 - RSE デーモン (RSED)。これは接続セットアップを管理します。また、単一サーバー・モードでの実行を担当します。そのために、RSE デーモンは RSE スレッド・プール (RSEDx) と呼ばれる子プロセスを 1 つ以上作成します。
 - RSE サーバー。これは個々のクライアント要求を処理します。RSE サーバーは、RSE スレッド・プール内のスレッドとしてアクティブになります。
- TSO コマンド・サービス (TSO cmd) は、TSO および ISPF コマンドに、バッチに似たインターフェースを提供します。
- JES ジョブ・モニター (JMON) は、JES に関連したすべてのサービスを提供します。
- 共通アクセス・リポジトリ・マネージャー (CARMA) は、CA Endevor などの Software Configuration Manager (SCM) と対話するためのインターフェースを提供します。
- SCLM Developer Toolkit (SCLMDT) は、SCLM を拡張し、SCLM と対話するためのインターフェースを提供します。
- Application Deployment Manager (ADM) は、CICS に関連したさまざまなサービスを提供します。
- ほかに、Developer for System z 自体または相互前提条件のソフトウェアで各種のサービスが提供されています。

前の段落とリストで説明したのは、RSE に割り当てられている中心的な役割です。わずかな例外を除き、クライアント通信はすべて RSE を経由します。これにより、クライアント/ホスト通信に使用されるポートの数が限定されるため、セキュリティに関連したネットワーク・セットアップが容易になります。

クライアントからの接続とワークロードを管理するために、RSE は、スレッド・プーリング・アドレス・スペースを制御するデーモン・アドレス・スペースから構成されています。デーモンは接続と管理のためのフォーカル・ポイントとして機能し、スレッド・プールはクライアント・ワークロードを処理します。rsed.envvars 構成ファイルに定義されている値と実際のクライアント接続数に基づいて、デーモンは複数のスレッド・プール・アドレス・スペースを開始することができます。

Java アプリケーションとしての RSE

z/OS UNIX プロセス



Java ストレージの使用

システム - 共用
システム - 専用
コード (z/OS UNIX、Java、RSE)
Java ヒープ
未使用

JOBNAME	状況	PID	PPID	コマンド
RSED	FILE SYS KERNEL WAIT	50331904	1	BPXBATSL
RSED	WAITING FOR CHILD	67109114	50331904	/bin/sh...
RSED	FILE SYS KERNEL WAIT	50331949	67109114	java...
RSED8	WAITING FOR CHILD	307	50331949	/bin/sh...
RSED8	FILE SYS KERNAL WAIT	308	307	java...

図 2. Java アプリケーションとしての RSE

図 2 は、RSE によるリソースの使用（プロセスとストレージ）を基本的な図で示しています。

RSE は Java™ アプリケーションであるため、z/OS UNIX 環境でアクティブになります。このため、異なるホスト・プラットフォームへの移植が容易であり、同じく Java アプリケーションである (Eclipse フレームワーク・ベース) Developer for System z クライアントと簡単に通信することができます。したがって、z/OS UNIX および Java の仕組みに関する基本的な知識があると、Developer for System z を理解するうえで非常に役立ちます。

z/OS UNIX では、プログラムが PID (プロセス ID) で識別されるプロセス内で実行されます。各プログラムはそのプログラム専用のプロセス内でアクティブになるため、別のプログラムを呼び出すと新しいプロセスが作成されます。プロセスを開始したプロセスは、PPID (親 PID) で参照され、新しいプロセスは子プロセスと呼ばれます。子プロセスは同じアドレス・スペース内で実行することも、新しいアドレス・スペースで spawn (作成) することもできます。同じアドレス・スペースで新しいプロセスを実行することは、TSO でコマンドを実行することに相当し、新しいアドレス・スペースでプロセスを spawn することは、バッチ・ジョブを実行依頼することと似ています。

プロセスは、単一スレッドの場合とマルチスレッドの場合があるので注意してください。マルチスレッド・アプリケーション (RSEなど) では、個別のアドレス・スペースのとき (より少ないオーバーヘッドによる) と同様に、個々のスレッドがシステム・リソースを得ようと競合します。

上記のプロセスの説明を 5 ページの図 2 の RSE サンプルに対応付けると、次のような流れになります。

1. RSED タスクが開始されると、このタスクが BPXBATSL を実行し、それによって z/OS UNIX が起動され、シェル環境が作成されます - PID 50331904。
2. このプロセスで rsed.sh シェル・スクリプトが実行され、それが別のプロセス (/bin/sh) で稼働します - PID 67109114。
3. このシェル・スクリプトは、rsed.envvars で定義されている環境変数を設定し、RSE デーモンを始動するために必要なパラメーターで Java を実行します - PID 50331949。
4. RSE デーモンが子プロセス (RSED8) で新しいシェルの spawn します - PID 307。
5. このシェルでは、rsed.envvars で定義されている環境変数が設定され、RSE スレッド・プールを始動するために必要なパラメーターで Java が実行されます - PID 308。

RSE は、31 ビットまたは 64 ビットのアドレッシング・モードで実行できるため、それぞれでストレージ限度も異なります。31 ビット・モードで使用可能なストレージの限度は 2 GB であり、64 ビット・モードでは、SYS1.PARMLIB で指定しない限り、限度はありません。

RSE などの Java アプリケーションは、ストレージを直接割り振るのではなく、Java メモリ管理サービスを使用します。これらのサービスは、ストレージの割り振り、ストレージの解放、およびガーベッジ・コレクションと同様に、Java ヒープの限度内で機能します。ヒープの最小サイズと最大サイズは、Java 始動時に (暗黙的または明示的に) 定義されます。64 ビット・モードで実行中の場合、Java は、境界より下のスペースを解放して、2 GB 境界より上のヒープを割り振ろうとします。

つまり、使用可能なアドレス・スペース・サイズを最大限に活用することは、z/OS 用に不定量 (アクティブなスレッド数に依存します) のシステム制御ブロックを保管できる余裕を確保しながら、大きなヒープ・サイズを定義するというバランスを考慮した両立案になります。

タスク所有者

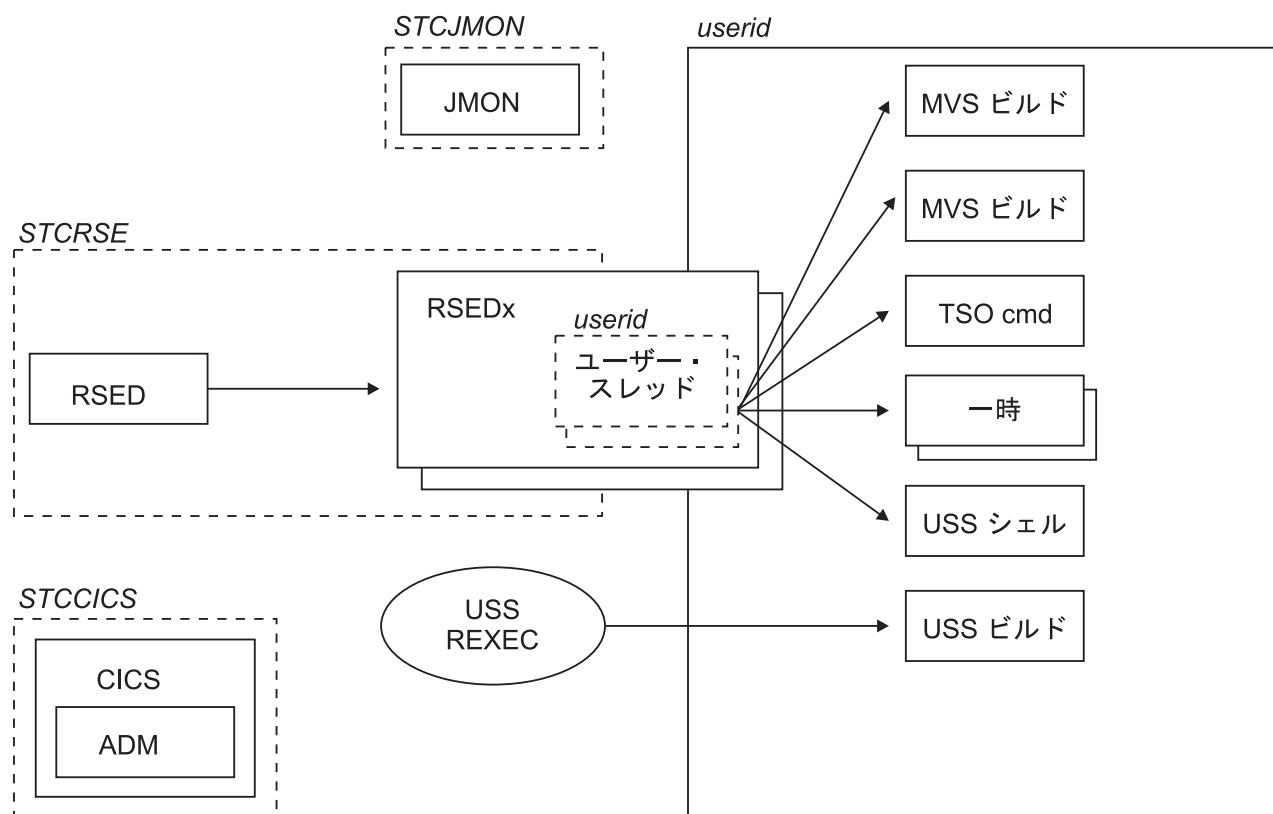


図3. タスク所有者

図3 は、Developer for System z のさまざまなタスクで使用するセキュリティ資格情報の所有者の基本的概要を示しています。

タスクの所有権は、2 つの部分に分けることができます。開始タスクは、ご使用のセキュリティ・ソフトウェアで開始タスクに割り当てられているユーザー ID が所有します。それ以外のすべてのタスク (RSE スレッド・プール (RSEDx) は例外) は、クライアント・ユーザー ID が所有します。

図3 は、Developer for System z の開始タスク (JMON および RSED)、およびサンプルの開始タスクと Developer for System z が通信するシステム・サービスを示しています。Application Deployment Manager (ADM) が CICS 領域内でアクティブになっています。USS REXEC タグは、z/OS UNIX REXEC (または SSH) サービスを表します。

RSE デーモン (RSED) は、クライアント要求を処理するために RSE スレッド・プール・アドレス・スペース (RSEDx) を 1 つ以上作成します。各 RSE スレッド・プールは、複数のクライアントをサポートし、RSE デーモンと同じユーザーによって所有されます。各クライアントには、スレッド・プール内に専用のスレッドがあり、これらのスレッドはクライアント・ユーザー ID が所有します。

クライアントが実行するアクションによっては、1 つ以上の追加のアドレス・スペース (いずれもクライアント・ユーザー ID が所有) を開始して要求されたアクションを実行できます。これらのアドレス・スペースにできるのは、MVS バッチ・ジョブ、APPC トランザクション、または z/OS UNIX 子プロセスです。z/OS UNIX 子

プロセスは、z/OS UNIX イニシエーター (BPXAS) 内でアクティブとなり、JES で開始タスクとして表示されることに注意してください。

これらのアドレス・スペースの作成は、ほとんどの場合、スレッド・プール内のユーザー・スレッドによって、直接的に、あるいは ISPF などのシステム・サービスを使用してトリガーされます。ただし、アドレス・スペースはサード・パーティーが作成する可能性もあります。例えば、File Manager は、Developer for System z に代わって処理する必要があるデータ・セット (またはメンバー) ごとに、新しいアドレス・スペースを開始します。z/OS UNIX でビルドを開始する際には、z/OS UNIX REXEC または SSH が関与します。

ユーザー固有のアドレス・スペースは、タスクが完了するか、または非アクティブ・タイマーの期限が切れると終了します。開始タスクはアクティブなままとなります。7 ページの図 3 に示されているアドレス・スペースは、表示の対象となるほど長くシステムに残ります。ただし、z/OS UNIX の設計仕様のために、存続期間の短い一時的なアドレス・スペースもいくつか存在することに注意してください。

接続のフロー

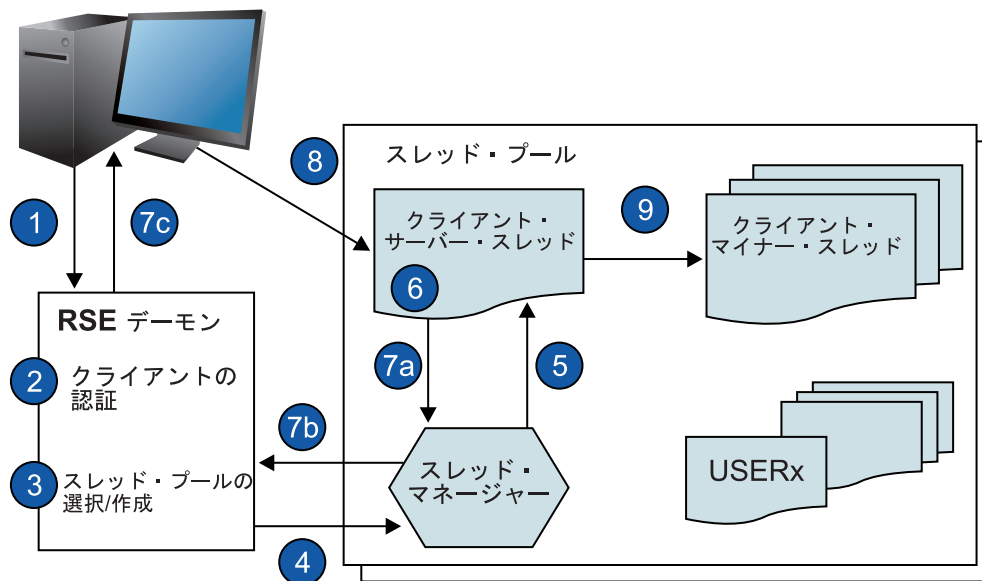


図 4. 接続のフロー

図 4 は、クライアントが Developer for System z を使用してホストに接続する仕組みの概要図です。ここでは、PassTicket の使い方についても簡単に説明します。

1. クライアントはデーモン（ポート 4035）にログオンします。
2. RSE デーモンは、クライアントから提示された資格情報を使用してクライアントを認証します。
3. RSE デーモンは、既存のスレッド・プールを選択します。すべてのプールが満杯である場合は、新しいスレッド・プールを開始します。
4. RSE デーモンは、スレッド・プールにクライアント・ユーザー ID を渡します。

5. スレッド・プールは、クライアント・ユーザー ID と認証用の PassTicket を使用して、クライアント固有の RSE サーバー・スレッドを作成します。
6. クライアント・サーバー・スレッドは、今後のクライアント通信に使用するポートにバインドします。
7. クライアント・サーバー・スレッドは、クライアントの接続先となるポート番号を返します。
8. クライアントは、RSE デーモンとの接続を解除し、提供されたポート番号に接続します。
9. クライアント・サーバー・スレッドは、他のユーザー固有のスレッド (マイナー) を開始します。その際、常にクライアント・ユーザー ID と認証用の PassTicket を使用します。これらのスレッドから、クライアントが要求したユーザー固有のサービスが提供されます。

上記の説明は、RSE のスレッド指向の設計を示しています。ユーザー単位でアドレス・スペースを開始するのではなく、複数のユーザーが単一のスレッド・プール・アドレス・スペースでサービスされます。スレッド・プール内では、各マイナー (ユーザー固有のサービス) が、ユーザーのセキュリティー・コンテキストが割り当てられたそのマイナー専用のスレッドでアクティブとなるため、セキュアなセットアップが確保されます。この設計は、リソース使用量を制限しながら多数のユーザーに対応しますが、各クライアントが複数のスレッド (実行されるタスクによっては 17 以上) を使用することを実際には意味しています。

ネットワークの観点では、Developer for System z はパッシブ・モードの FTP と同じように動作します。クライアントはフォーカル・ポイント (RSE デーモン) に接続し、その接続をドロップして、フォーカル・ポイントから提供されたポート番号に再接続します。この 2 番目の接続に使用されるポートの選択を制御するロジックを以下に示します。

1. クライアントがサブシステム・プロパティー・タブでゼロ以外のポート番号を指定した場合、RSE サーバーはそのポートをバインドに使用します。このポートが使用不可であると、接続は失敗します。
2. `rsed.envvars` に `_RSE_PORTRANGE` が指定されている場合、RSE サーバーはこの範囲から選択したポートにバインドします。使用可能なポートがない場合、接続は失敗します。RSE サーバーはクライアント接続の期間中に、ポートを独占的に必要とすることはありません。他の RSE サーバーがそのポートにバインドできないのは、(サーバーの) バインドから (クライアントの) 接続までの間だけです。つまり、ほとんどの接続が範囲内の最初のポートを使用し、同じ範囲内の残りのポートは、同時に複数のログオンが発生したときのバッファになる、ということです。
3. 制限が設定されていなければ、RSE サーバーはポート 0 にバインドします。その結果、TCP/IP によってポート番号が選択されます。

認証を必要とするすべての z/OS サービスに PassTicket が使用されるので、Developer for System z は、パスワードを保管したりユーザーに毎回パスワードを要求したりすることなく、これらのサービスを任意に呼び出すことができます。また、すべての z/OS サービスに PassTicket を使用することで、ログオン時にワンタイム・パスワードや X.509 証明書などの代替認証方式を使用することもできます。

統合デバッガー

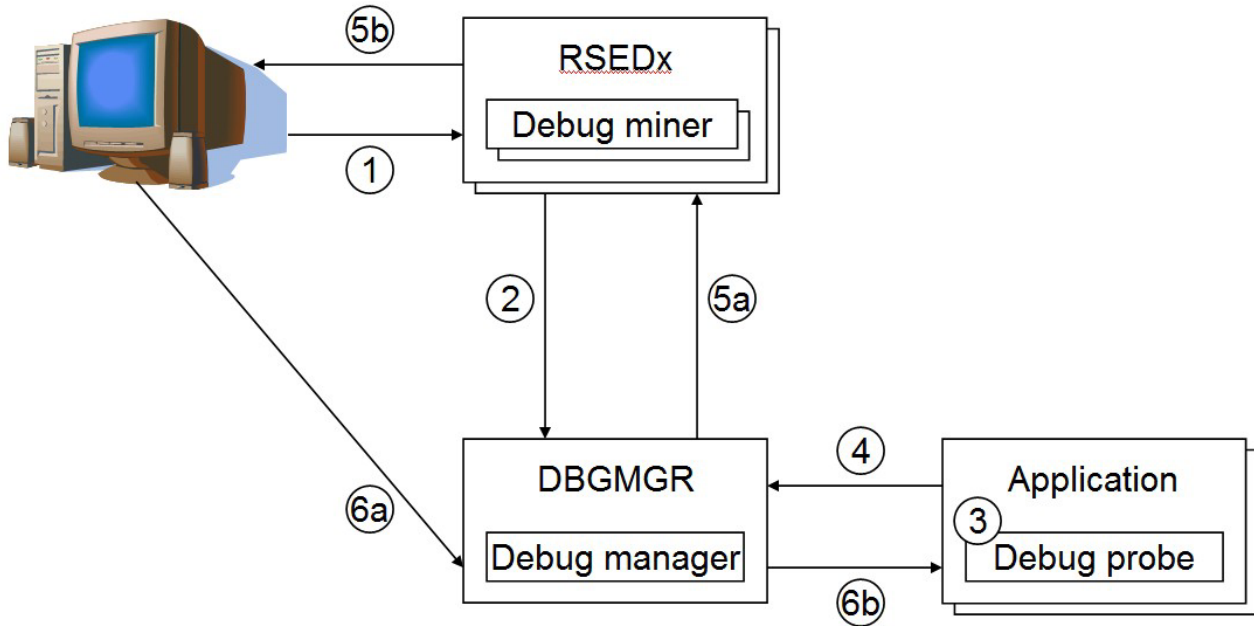


図 5. 統合デバッガー

統合デバッガーは、さまざまなアプリケーションのデバッグに使用されます。図 5 に、Developer for System z クライアントがアプリケーションのデバッグをどのように行えるかに関する概念図を示します。

1. クライアントは、標準 Developer for System z ホスト・ログオンを使用してホストに接続します。
2. ログオンの一部として、デバッグ・マイナーがユーザーをデバッグ・マネージャーに登録します。デバッグ・マネージャーは DBGMGR 開始タスクでアクティブになっています。
3. アプリケーションがデバッグを必要とする標識付きで開始した場合、言語環境プログラム (Language Environment® (LE)) がデバッグ・プローブを呼び出します。
4. デバッグ・プローブはデバッグ・マネージャーに登録します。
5. デバッグ・マイナーを使用して、デバッグ・マネージャーはこのデバッグ・セッションを受け取るユーザーの Developer for System z クライアントに通知します。ユーザーがこの時点で登録されていない場合、デバッグ・セッションは休止し、ユーザーがデバッグ・マネージャーに登録されるまで待機します。
6. クライアント内のデバッグ・エンジンはデバッグ・マネージャーに連絡を取り、今度はデバッグ・マネージャーがデバッグ・エンジンとデバッグ・プローブ間で行き来するデータの受け渡しを行います。

CARMA

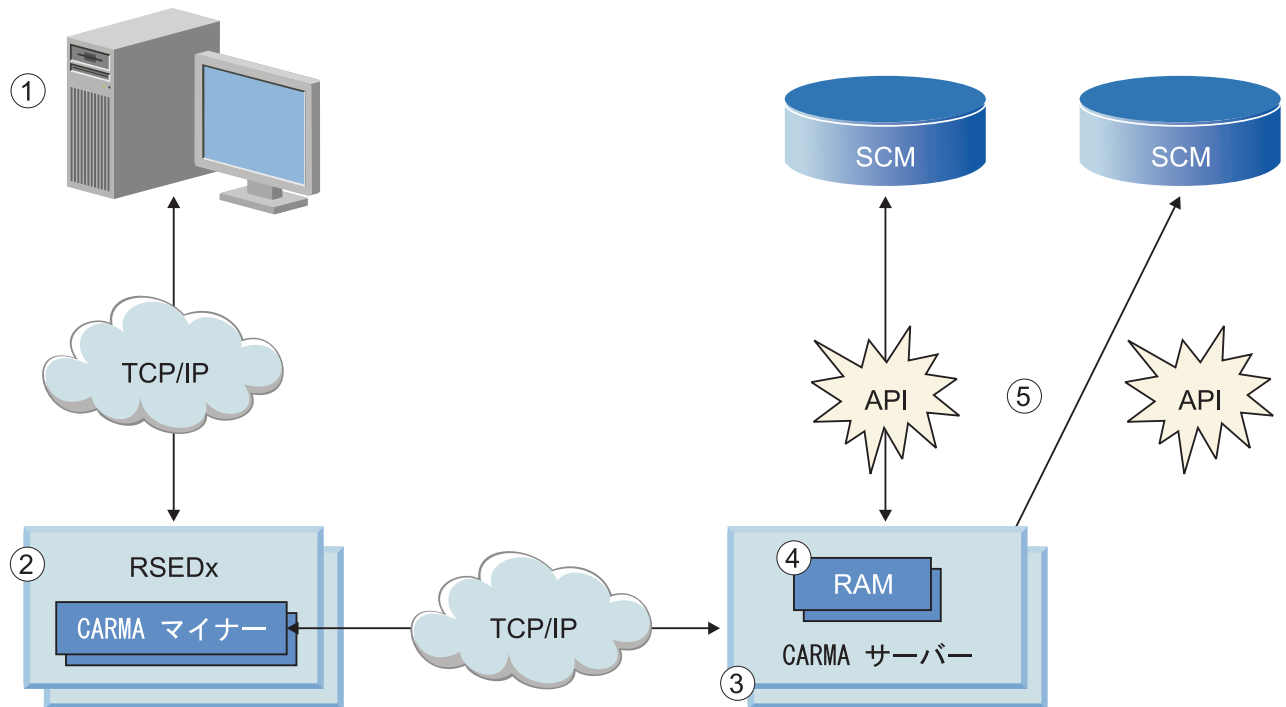


図 6. CARMA フロー

CARMA (Common Access Repository Manager) は、ホスト・ベースの Software Configuration Manager (SCM) (CA Endeavor® SCM など) にアクセスするために使用されます。図 6 は、サポートされたホスト・ベースの Software Configuration Manager (SCM) に Developer for System z のクライアントがアクセスする仕組みの概要図です。

1. クライアントには、共通アクセス・リポジトリ・マネージャー (CARMA) プラグインがあります。
2. CARMA プラグインは CARMA マイナーと通信します。これは RSE スレッド・プール (RSEDx) 内のユーザー固有のスレッドとしてアクティブになっています。この通信は、既存の RSE 接続経由で行われます。
3. クライアント要求が SCM にアクセスすると、CARMA マイナーは TCP/IP ポートにバインドし、ポート番号を始動引数として、ユーザー固有の CARMA サーバーを始動します。すると CARMA サーバーは、このポートに接続し、このパスをクライアントとの通信に使用します。ホスト・ベースの SCM は、単一ユーザーのアドレス・スペースを想定してサービスにアクセスすることに注意してください。そのため、CARMA はユーザーごとに CARMA サーバーを始動する必要があります。複数のユーザーをサポートする単一サーバーを作成することはできません。
4. CARMA サーバーは、要求された SCM をサポートする Repository Access Manager (RAM) をロードします。
5. この RAM は、特定の SCM との対話の技術的な詳細を処理し、クライアントへの共通インターフェースを提供します。

CARMA 構成ファイル

Developer for System z は CARMA サーバーを始動する複数の方式をサポートしています。それぞれの方式には利点と欠点があります。Developer for System z も、複数の Repository Access Manager (RAM) を提供します。これらは実動 RAM とサンプル RAM という 2 つのグループに分けられます。事前構成されたセットアップとして、RAM とサーバー始動方式のさまざまな組み合わせが可能です。

すべてのサーバー始動方式は共通の構成ファイル `CRASRV.properties` を共有します。このファイルは (特に) どの始動方式を使用するかを指定します。

CRASTART

「CRASTART」方式は、CARMA サーバーを RSE 内のサブタスクとして始動します。この方式では、CARMA サーバーを始動するために必要なデータ・セット割り振りとプログラム呼び出しを別個の構成ファイルで定義し、その構成ファイルを使用するので、非常に柔軟なセットアップが可能です。この方式では最良のパフォーマンスが得られ、使用するリソースも最少で済みますが、モジュール CRASTART を LPA 内に配置する必要があります。

RSE は、ロード・モジュール CRASTART を呼び出します。これは `crastart*.conf` の定義を使用してバッチ TSO と ISPF コマンドを実行するのに有効な環境を作成します。Developer for System z はこの環境を使用して、CARMA サーバー CRASERV を稼働させます。Developer for System z は複数の `crastart*.conf` ファイルを提供します。これらのファイルは、それぞれ特定の RAM 用に事前構成されています。

バッチ実行依頼

「バッチ実行依頼」方式は、ジョブを実行依頼することによって CARMA サーバーを始動します。これが、提供されたサンプル構成ファイルで使用するデフォルトの方式です。この方式の利点は、ジョブ出力内で CARMA ログに簡単にアクセスできることです。また、開発者自身が保守する開発者ごとのカスタム・サーバー JCL を使用できます。ただし、この方式では、CARMA サーバーを始動した開発者ごとに 1 つずつ JES イニシエーターが使用されます。

RSE は `CLIST CRASUB*` を呼び出します。これは次に組み込み JCL を実行依頼して、バッチ TSO と ISPF コマンドを実行するのに有効な環境を作成します。Developer for System z はこの環境を使用して、CARMA サーバー CRASERV を実行します。Developer for System z は、複数の `CRASUB*` メンバーを提供します。これらのメンバーは、それぞれ特定の RAM 用に事前構成されています。

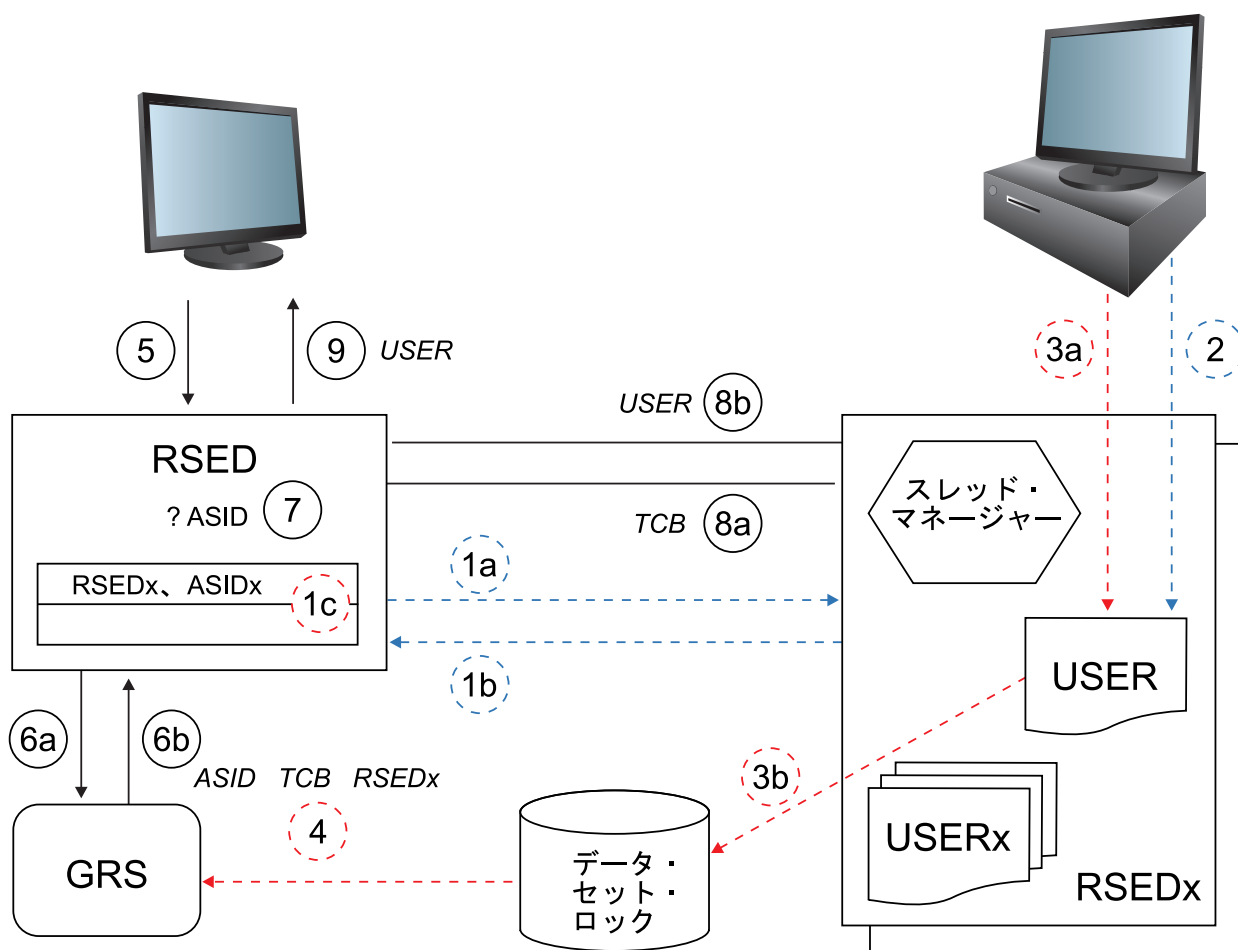


図7. データ・セットのエンキュー判別のフロー

図 7 は、データ・セット・ロックを所有する Developer for System z クライアントを RSE デーモンが判別する仕組みの概要図です。

1. RSE デーモン (RSED) はスレッド・プール (RSEDx) を作成します。開始完了を確認するため、スレッド・プールが RSE デーモンにそのアドレス・スペース ID (ASID) を報告します。RSE デーモンは、このスレッド・プールを追跡するために作成された制御ブロックにその ASID を保管します。
2. クライアントがログオンすると、スレッド・プール (RSEDx) 内にユーザー固有の RSE サーバー・スレッド (USER) が作成されます。各スレッドには、固有のタスク制御ブロック (TCB) ID があります。
3. クライアントが編集モードでデータ・セットをオープンすると、RSE サーバーはそのデータ・セットに排他ロック (エンキュー) を設定するように指示を受けます。
4. システムは、要求側の ASID、TCB、およびタスク名 (RSEDx) をエンキュー・プロセスの一部として登録します。この情報は、グローバル・リソースの逐次化 (GRS) キューに保管されます。

5. オペレーターが、RSE デーモンにデータ・セットのロック状況を照会します。
6. RSE デーモンは GRS キューをスキャンし、データ・セットがロックされているかどうかを確認し、ロック所有者の ASID、TCB、およびタスク名を取り出します。
7. 取り出した ASID は異なるスレッド・プールの ASID と比較されます。
8. RSE デーモンは ASID を所有しているスレッド・プールに、どのユーザーが TCB を所有しているかを判別するよう求めます。
9. 一致が確認された場合は、関連するクライアント・ユーザー ID が要求側に返されます。そうでない場合は、GRS から取り出されたタスク名が返されます。

複数のユーザーが単一のスレッド・プール・アドレス・スペースに割り当てられる、Developer for System z のシングル・サーバー・セットアップでは、z/OS が「**DISPLAY GRS,RES=(*,dataset*)**」オペレーター・コマンドを使用してデータ・セットまたはメンバーに設定されているロックの所有者を追跡できません。システム・コマンドはアドレス・スペース・レベル、つまりスレッド・プールで停止します。

この問題に対処する方法として、「ホスト構成ガイド」(SC88-5663) の『オペレーター・コマンド』で説明されているように、Developer for System z は「**MODIFY rshed APPL=DISPLAY OWNER,DATASET=dataset**」オペレーター・コマンドを提供します。オペレーター・コマンドは、RSE ユーザーがデータ・セットおよびメンバーに設定したすべてのロック、および ISPF などの他の製品が設定したロックを解決できます。

ロックの解放

通常的环境では、データ・セットまたはメンバーは、クライアントが編集モードでオープンしたときにロックされ、クライアントが編集セッションを終了したときに解放されます。

一定のエラー条件によって、このメカニズムが設計どおりに機能しなくなることがあります。この場合は、ロックを保持しているユーザーを RSE の **modify cancel** オペレーター・コマンドで取り消すことができます。これについては「ホスト構成ガイド」(SC88-5663)の『オペレーター・コマンド』で説明しています。このユーザーに属しているアクティブなデータ・セット・ロックが、プロセスの中で解放されます。

z/OS UNIX ディレクトリー構造

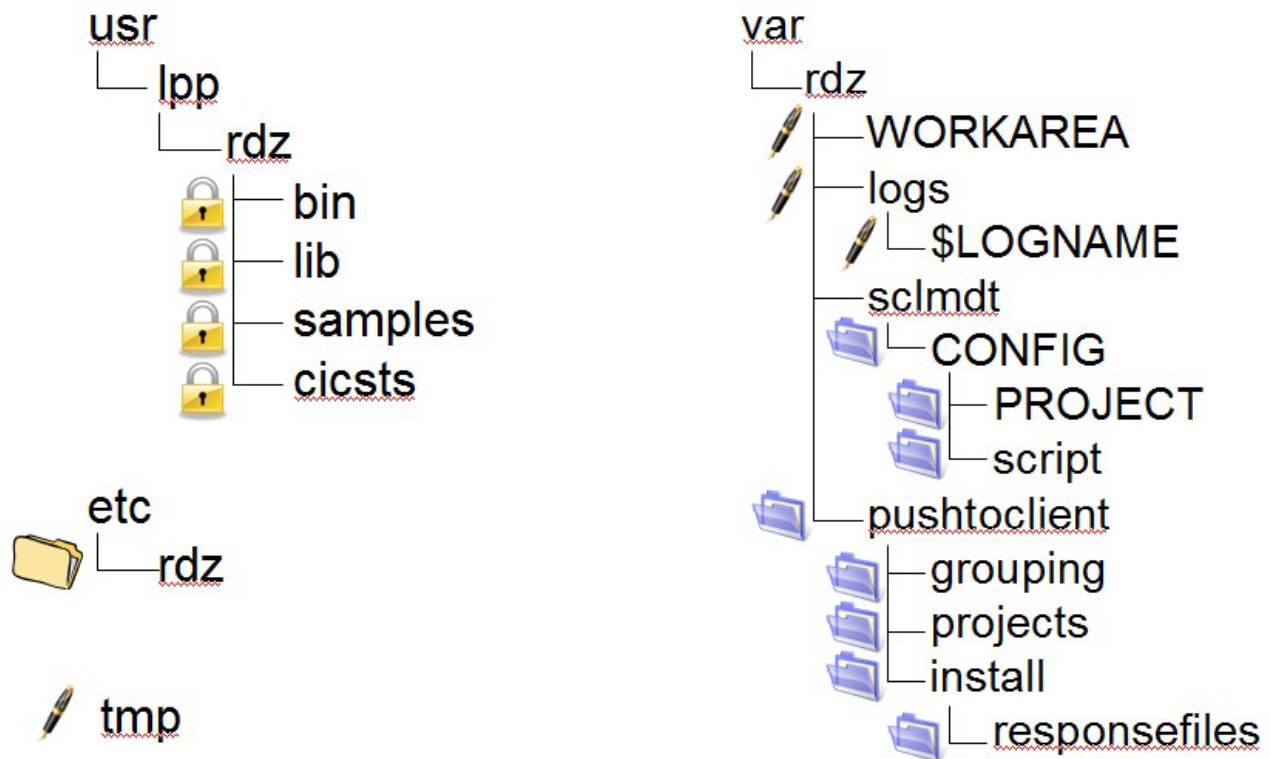


図 8. z/OS UNIX ディレクトリー構造

図 8 は、Developer for System z が使用する z/OS UNIX ディレクトリーの概要を示しています。以下のリストでは、Developer for System z が関与する各ディレクトリー、ロケーションの変更方法、および内部のデータを保守する当事者について説明します。

- `/usr/lpp/rdz/` は、Developer for System z 製品コードのルート・パスです。実際のロケーションは、RSED 開始タスク (変数 `HOME`) で指定されます。この中のファイルは、SMP/E が保守します。
- `/etc/rdz/` には、RSE とマイナーに関連する構成ファイルが保持されます。実際のロケーションは、RSED 開始タスク (変数 `CNFG`) で指定されます。この中のファイルは、システム・プログラマーが保守します。
- `/tmp/` は、ISPF の TSO/ISPF クライアント・ゲートウェイと各種マイナーが一時データを保管するために使用します。ここに出力を保管する IVP もあります。この中のファイルは、ISPF、マイナー、および IVP が保守します。このロケーションは、`rsed.envvars` の `TMPDIR` 変数でカスタマイズできます。これは Java ダンプ・ファイルのデフォルト・ロケーションでもあります。このロケーションは、`rsed.envvars` の `_CEE_DUMPTARG` 変数でカスタマイズできます。

注: 各クライアントが一時ファイルを作成できるようにするには、`/tmp/` に許可ビット・マスク `777` が必要です。

- `/var/rdz/WORKAREA/` は、ISPF の TSO/ISPF クライアント・ゲートウェイおよび SCLMDT が、z/OS UNIX と MVS ベースのアドレス・スペース間でデータを転

送するために使用します。実際のロケーションは、rsed.envvars (変数 CGI_ISPWORK) で指定されます。この中のファイルは、ISPF および SCLMDT が保守します。

注: 各クライアントが一時ファイルを作成できるようにするには、/var/rdz/WORKAREA/ に許可ビット・マスク 777 が必要です。

- /var/rdz/logs/ には、RSE デーモンおよび RSE スレッド・プール・サーバーのログが保持されます。実際のロケーションは、rsed.envvars (変数 daemon.log) で指定されます。この中のファイルは、RSE が保守します。
- /var/rdz/logs/\$LOGNAME/ には、RSE サーバーおよびマイナーのユーザー固有ログが保持されます。実際のロケーションは、rsed.envvars (変数 user.log および DSTORE_LOG_DIRECTORY) で指定されます。この中のファイルは、RSE およびマイナーが保守します。

注: 各クライアントが独自の \$LOGNAME ディレクトリーを作成してユーザー固有のログ・ファイルを保管できるようにするには、/var/rdz/logs/ に許可ビット・マスク 777 が必要です。

- /var/rdz/sclmdt/CONFIG/ には、汎用の SCLMDT 構成ファイルが保持されます。実際のロケーションは、rsed.envvars (変数 SCLMDT_CONF_HOME) で指定されます。この中のファイルは、SCLM 管理者が保守します。
- /var/rdz/sclmdt/CONFIG/PROJECT/ には、SCLMDT プロジェクト構成ファイルが保持されます。実際のロケーションは、rsed.envvars (変数 SCLMDT_CONF_HOME) で指定されます。この中のファイルは、SCLM 管理者が保守します。
- /var/rdz/sclmdt/CONFIG/script/ には、他の製品が使用する SCLMDT 関連のスク립トが保持されます。実際のロケーションはどこにも指定されません。この中のファイルは、SCLM 管理者が保守します。
- /var/rdz/pushtoclient/ は、クライアントの構成ファイル、クライアント製品の更新情報、およびホスト・ベースのプロジェクト情報 (ホストへの接続時にクライアントに送信される) を保持しています。実際のロケーションは、pushtoclient.properties (変数 pushtoclient.folder) で指定されます。この中のファイルは、Developer for System z クライアント管理者が保守します。
- /var/rdz/pushtoclient/grouping/ は、グループ固有のクライアント構成ファイル、クライアント製品の更新情報、およびホスト・ベースのプロジェクト情報 (ホストへの接続時にクライアントに送信される) を保持しています。実際のロケーションは、pushtoclient.properties 内で指定されます (変数 pushtoclient.folder にサフィックス /grouping を付加する)。この中のファイルは、Developer for System z クライアント管理者が保守します。
- /var/rdz/pushtoclient/projects/ には、ホスト・ベースのプロジェクト定義ファイルが保持されます。実際のロケーションは、/var/rdz/pushtoclient/keymapping.xml で指定されます。これは、Developer for System z クライアント管理者が作成し、保守します。この中のファイルは、プロジェクト・マネージャーまたは主任開発者が保守します。
- /var/rdz/pushtoclient/install/ には、ホストへの接続時にクライアント製品のバージョンを更新するために使用される構成ファイルが保持されます。実際のロケーションは、/var/rdz/pushtoclient/keymapping.xml で指定されます。これは、Developer for System z クライアント管理者が作成し、保守します。この中のファイルは、クライアント管理者が保守します。

- `/var/rdz/pushtoclient/install/responsefiles/` には、ホストへの接続時にクライアント製品のバージョンを更新するために使用される構成ファイルが保持されます。実際のロケーションは、`/var/rdz/pushtoclient/keymapping.xml` で指定されます。これは、Developer for System z クライアント管理者が作成し、保守します。この中のファイルは、クライアント管理者が保守します。

非システム管理者の更新特権

`/var/rdz/pushtoclient/` 内のデータは、z/OS UNIX で多数の更新特権を持たない可能性のあるプロジェクト・マネージャーなどの非システム管理者によって保守されます。したがって、実際のだがセキュアなセットアップを確実に行うために、z/OS UNIX において、ファイル作成時にどのようにアクセス権限が設定されるのかを理解することが重要です。

UNIX 標準では、所有者、グループ、およびその他という 3 つのタイプのユーザーに対して権限を設定できることが定められています。読み取り権限、書き込み権限、および実行権限は、各タイプに対して個別に設定できます。

z/OS UNIX では、ファイル作成時に UID (ユーザー ID) および GID (グループ ID) が以下の値に設定されます。

- UID は、作成スレッドの実効 UID に設定されます。
- GID は、所有ディレクトリーの GID に設定されます。セキュリティー・プロファイル `FILE.GROUPOWNER.SETGID` が、`UNIXPRIV` クラスに定義されているのであれば、代わりにデフォルトで、作成スレッドの実効 GID が使用されます。詳細については、「*UNIX System Services 計画*」(GA88-8639) を参照してください。

各サイトでは、それぞれ独自のデフォルト・アクセス許可マスクを設定できますが、共通マスクは、所有者に読み取り権限と書き込み権限を許可し、グループとその他のユーザーに読み取り権限を許可します。

`/var/rdz/pushtoclient/` 内のデータは、`pushtoclient.properties` の `file.permission` ディレクティブに定義されたアクセス許可マスクを使用して作成されます。デフォルト値では、所有者とグループに対して読み取り権限と書き込み権限が許可され、その他のユーザーに読み取り権限が許可されます。すべてのユーザーが実行権限を持ちます。最終的なアクセス権限では、すべてのユーザーに対し読み取り権限と実行権限を許可し、データを保守する Developer for System z クライアント管理者にはさらに書き込み権限を許可する必要があります。

`/var/rdz/pushtoclient/projects/` 内のデータは、特定のアクセス許可マスクを使用せずに作成されます。最終的なアクセス権限では、すべてのユーザーに対し読み取り権限を許可し、データを保守するプロジェクト・マネージャーにはさらに書き込み権限を許可する必要があります。

便利なセキュリティー・コマンド

プロジェクト・マネージャーまたは Developer for System z クライアント管理者のグループが、これらのディレクトリー内のデータを確実に管理できるようにするために、セキュリティー・マネージャーは、そのプロジェクト・マネージャーやクライアント管理者に対して有効な OMVS セグメントを指定したグループを作成しなければならない場合があります。このグループは、できれば、関連するユーザー ID

のデフォルト・グループにします。次のサンプル RACF® コマンドについて詳しくは、「*Security Server RACF コマンド言語解説書*」(SA88-8617) を参照してください。

```
ADDGROUP RDZPROJ OMVS(GID(1200))
CONNECT IBMUSER GROUP(RDZPROJ)
ALTUSER IBMUSER DFLTGRP(RDZPROJ)
```

便利な z/OS UNIX コマンド

次のサンプル z/OS UNIX コマンドについて詳しくは、「*UNIX System Services コマンド解説書*」(SA88-8641) を参照してください。

- 次の z/OS UNIX の **ls** コマンドを使用して、ディレクトリー内のすべてのファイルを表示します。

```
ls -lR /var/rdz/pushtoclient/
```

- 次の z/OS UNIX の **chown** コマンドを使用して、ディレクトリーとその中の全ファイルの所有者を変更します。

```
chown -R IBMUSER /var/rdz/pushtoclient/
```

- 次の z/OS UNIX の **chgrp** コマンドを使用して、このグループをディレクトリーとその中の全ファイルに割り当てます。

```
chgrp -R RDZPROJ /var/rdz/pushtoclient/
```

- 次の z/OS UNIX の **chmod** コマンドを使用して、所有者とグループにディレクトリーとその中の全ファイルに対する書き込み権限を付与します。その他のユーザーは読み取り権限を持ちます。すべてのユーザーが実行権限を持ちます。

```
chmod -R 775 /var/rdz/pushtoclient/
```

サンプル・セットアップ

以下のシナリオでは、すべての開発プロジェクト・マネージャーと 3 人のチームが、Developer for System z クライアント管理者の役割を担います。

セキュリティ管理者は、既に、固有のグループ ID (1200) を持つデフォルト・グループ (RDZPROJ) をそのチームに割り当てています。そのユーザー ID には、(UID 0 のような) z/OS UNIX での特権は特にありません。セキュリティ管理者は、FILE.GROUPOWNER.SETGID プロファイルを定義していないので、z/OS UNIX は、新規ファイル作成時にディレクトリーのグループ ID を使用します。ユーザー ID IBMUSER (UID 0 およびデフォルト・グループ SYS1) は、ディレクトリー /var/rdz/pushtoclient を作成するためにシステム・プログラマーによって使用されました。

1. システム・プログラマーは、以下のように /var/rdz/pushtoclient 書き込み権限を所有者とグループに制限します。

```
# chmod 775 /var/rdz/pushtoclient
# ls -ld /var/rdz/pushtoclient
drwxrwxr-x  2 IBMUSER SYS1
/var/rdz/pushtoclient
```

注: カスタマイズのセットアップ時に使用した FEKSETUP ジョブは、既にこのステップを実行しています。

2. システム・プログラマーは、以下のように RDZPROJ を所有グループにします。

```
# chgrp RDZPROJ /var/rdz/pushtoclient
# ls -ld /var/rdz/pushtoclient
drwxrwxr-x  2 IBMUSER RDZPROJ
/var/rdz/pushtoclient
```

これで、/var/rdz/pushtoclient 書き込み権限をシステム・プログラマー (IBMUSER) とプロジェクト・マネージャー (RDZPROJ) に制限するのに必要なセットアップは終わりです。

第 2 章 セキュリティーに関する考慮事項

Developer for System z では、メインフレーム以外のワークステーション上にいるユーザーがメインフレームにアクセスできます。このため、接続要求の妥当性検査、ホストとワークステーション間のセキュアな通信の提供、およびアクティビティーの許可と監査が、製品構成の重要な側面となります。

Developer for System z サーバーとサービスが使用するセキュリティー・メカニズムは、それが存在するデータ・セットとファイル・システムがセキュアであることに依存しています。つまり、信頼されたシステム管理者のみがプログラム・ライブラリーと構成ファイルを更新できる状態でなければなりません。

この章では、以下のトピックについて説明します。

- 22 ページの『認証方式』
- 23 ページの『接続セキュリティー』
- 25 ページの『PassTicket の使用』
- 27 ページの『監査ロギング』
- 29 ページの『JES セキュリティー』
- 33 ページの『SSL/TLS 暗号化通信』
- 35 ページの『クライアント認証、X.509 証明書を使用した』
- 39 ページの『Port Of Entry (POE) 検査』
- 43 ページの『各種情報』
- 39 ページの『クライアント関数の変更』
- 41 ページの『クライアントへのプッシュの開発者グループ』
- 42 ページの『デバッグ・セキュリティー』
- 43 ページの『CICSTS セキュリティー』
- 44 ページの『SCLM セキュリティー』
- 44 ページの『Developer for System z 構成ファイル』
- 47 ページの『セキュリティー定義』

注: リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供し、次の 2 つの論理エンティティーから構成されています。

- RSE デーモン。これは接続セットアップを管理し、開始タスクとして開始されるか、長時間実行ユーザー・ジョブとして開始されます。
- RSE サーバー。これは個々のクライアント要求を処理し、RSE デーモンによって、1 つ以上の子プロセス内のスレッドとして開始されます。

Developer for System z の基本的な設計概念については、3 ページの『第 1 章 Developer for System z について』を参照してください。

認証方式

Developer for System z は、接続時にクライアントが提供するユーザー ID を認証するために、複数の方法をサポートしています。

- ユーザー ID およびパスワード
- ユーザー ID およびワンタイム・パスワード
- X.509 証明書

注: クライアントが提供した認証データは、初期の接続セットアップ時に 1 回だけ使用されます。ユーザー ID が認証された後、認証を必要とするすべてのアクションには、そのユーザー ID と自己生成された PassTicket が使用されます。

ユーザー ID およびパスワード

クライアントは接続時に、ユーザー ID とそれに一致するパスワードを提供します。そのユーザー ID とパスワードは、使用しているセキュリティ製品でのユーザーの認証に使用されます。

ユーザー ID およびワンタイム・パスワード

ワンタイム・パスワードは、固有のトークンを基に、サード・パーティー製品で生成できます。ワンタイム・パスワードでは、ユーザーが知らないうちに固有のトークンをコピーして使用することはできないため、セキュリティのセットアップが向上し、またパスワードをインターセプトしても、それは一回限り有効であるため役に立ちません。

クライアントは接続時に、ユーザー ID とワンタイム・パスワードを提供します。このパスワードは、サード・パーティーが提供するセキュリティ出口で、ユーザー ID の認証に使用されます。このセキュリティ出口では、通常の処理時に認証要求を満たすために使用された PassTicket を無視することが見込まれます。PassTicket は、ご使用のセキュリティ・ソフトウェアによって処理する必要があります。

X.509 証明書

サード・パーティーは、ユーザーの認証に使用できる 1 つ以上の X.509 証明書を提供できます。X.509 証明書を機密保護機能のあるデバイスに保管すると、セキュアなセットアップとユーザーにとっての操作性 (ユーザー ID やパスワードが不要であること) を同時に実現できます。

接続時に、クライアントは選択された証明書と選択された拡張 (オプション) を提供します。これを使用して、ご使用のセキュリティ製品でユーザー ID の認証が行われます。

注: この認証方式をサポートしているのは、RSE デーモン接続方式のみです。また、SSL (Secure Socket Layer) を使用可能にする必要があります。

JES ジョブ・モニターの認証

クライアント認証は、クライアントの接続要求の一環として、RSE デーモン (または REXEC/SSH) によって行われます。ユーザーが認証されると、JES ジョブ・モニターへの自動ログオンも含め、その後のすべての認証要求には、自己生成された PassTicket が使用されます。

JES ジョブ・モニターは、RSE によって提供されたユーザー ID と PassTicket の妥当性検査を行うためには、PassTicket の評価を許可されている必要があります。これは、以下のことを意味します。

- ロード・モジュール FEJJMON (デフォルトではロード・ライブラリー FEK.SFEKAUTH に置かれています) に APF 許可があることが必要です。
- RSE と JES ジョブ・モニターの両方が、同じアプリケーション ID (APPLID) を使用する必要があります。デフォルトでは、どちらのサーバーも FEKAPPL を APPLID として使用しますが、これは変更でき、そのためには、RSE の場合は rsed.envvars 内で、JES ジョブ・モニターの場合は FEJJCENFG 内で、それぞれ APPLID ディレクティブを使用します。

注: 以前のクライアント (バージョン 7.0 以前) は、JES ジョブ・モニターと直接通信します。これらの接続では、ユーザー ID とパスワードによる認証方式のみがサポートされています。

デバッグ・マネージャーでの認証

クライアント認証は、クライアントの接続要求の一環として、RSE デーモン (または REXEC/SSH) によって行われます。ユーザーが認証されると、デバッグ・マネージャーへの自動ログオンも含め、その後のすべての認証要求には、自己生成された PassTicket が使用されます。

デバッグ・マネージャーが、RSE から提供されたユーザー ID と PassTicket を妥当性検査するためには、PassTicket を評価できる必要があります。これは、ロード・モジュール AQEZPCM (デフォルトではロード・ライブラリー FEK.SFEKAUTH 内にある) に APF 許可が必要であることを意味します。

接続セキュリティー

さまざまなレベルの通信セキュリティーが RSE によってサポートされており、RSE は、クライアントと大部分の Developer for System z サービスとの間の通信を制御します。

- 外部 (クライアント/ホスト) 通信を、指定したポートだけに制限できます。このフィーチャーは、デフォルトでは使用不可に設定されます。
- 外部 (クライアント/ホスト) 通信を SSL または TLS を使用して暗号化できます。このフィーチャーは、デフォルトでは使用不可に設定されます。
- 信頼できる TCP/IP アドレスのみにホスト・アクセスを許可できるようにするために、Port Of Entry (POE) 検査を使用できます。このフィーチャーは、デフォルトでは使用不可に設定されます。

一部のオプション Developer for System z サービスでは、分離した外部 (クライアント/ホスト) 通信パスを使用します。

- 統合デバッガー通信は TLS を使用して暗号化できます。
- Application Deployment Manager 通信は、Web サービス・インターフェースを使用する場合、SSL を使用して暗号化できます。

Developer for System z は一部のサービスの提供を、TN3270 サーバーなどのサーバ・パーティー製品に依存します。接続セキュリティー・オプションについては、関連する製品資料を参照してください。

指定したポートのみに限定した外部通信

システム・プログラマーは、RSE サーバーがクライアントと通信できるポートを指定できます。デフォルトでは、使用可能な任意のポートが使用されます。このポート範囲は、RSE デーモン・ポートとは関係ありません。

ポートの使用法を理解しやすいように、以下で RSE の接続プロセスを簡単に説明します。

1. クライアントは、ホスト・ポート 4035、RSE デーモンに接続します。
2. RSE デーモンは、RSE サーバー・スレッドを作成します。
3. RSE サーバーは、クライアントが接続するホスト・ポートを開きます。このポートの選択はユーザーが構成でき、クライアント上でサブシステム・プロパティ・タブによって構成するか (これはお勧めできません)、rsed.envvars 内の `_RSE_PORTRANGE` 定義を通じて構成します。
4. RSE デーモンは、クライアントにポート番号を返します。
5. クライアントは、ホスト・ポートに接続します。

注: このプロセスは、「ホスト構成ガイド」(SC88-5663)の「『(オプション) REXEC (または SSH) の使用』」で説明されている REXEC/SSH を使用した (オプションの) 代替接続方式とほとんど同じです。

SSL または TLS を使用した通信暗号化

RSE を通過するすべての外部 Developer for System z データ・ストリームを、Secure Sockets Layer (SSL) またはトランスポート層セキュリティー (TLS) を使用して暗号化できます。暗号化通信の使用は、33 ページの『SSL/TLS 暗号化通信』に説明されているように、ssl.properties 構成ファイル内の設定によって制御されます。rsed.envvars の `_RSE_JAVA_OPTS` ディレクティブの `DSTORE_SSL_ALGORITHM` 変数によって、SSL とその後継の TLS を暗号化方式として選択できます。詳しくは、「ホスト構成ガイド」(SC88-5663) の『`_RSE_JAVA_OPTS` での追加 Java 始動パラメーターの定義』を参照してください。

クライアントの統合デバッガー・エンジンは、ホスト上のデバッグ・マネージャーに接続します。SSL または TLS の使用は Application Transparent TLS (AT-TLS) ポリシーによって制御されます。

クライアント上の Host Connect Emulator は、ホスト上の TN3270 サーバーに接続します。SSL または TLS の使用は TN3270 によって制御されます。これについては、「Communications Server IP 構成ガイド」(SC88-8926) に説明があります。

z/OS UNIX サブプロジェクトのリモート (ホスト・ベースの) アクションでは、ホスト上の REXEC サーバーまたは SSH サーバーを使用します。SSH 通信は常に SSL を使用して暗号化します。

Application Deployment Manager クライアントは、CICS TS Web サービスまたは RESTful インターフェースを使用して、Application Deployment Manager ホスト・サービスを起動します。SSL の使用は、CICS TS によって制御されます。これについては、「*RACF Security Guide for CICS TS*」に説明があります。

Port Of Entry 検査

Developer for System z は Port Of Entry (POE) 検査をサポートしています。これにより、ホストは信頼できる TCP/IP アドレスにのみアクセスできます。POE の使用は、39 ページの『Port Of Entry (POE) 検査』で説明されているように、セキュリティ・ソフトウェア内の特定のプロファイルの定義と、`rsed.envvars` 内の `enable.port.of.entry` ディレクティブによって制御されます。

POE をアクティブにすると、POE 検査をサポートしている他の TCP/IP アプリケーション (INETD など) に影響が出ることに注意してください。

PassTicket の使用

ログオン後、PassTicket を使用して RSE サーバー内のスレッドのセキュリティが確立されます。このフィーチャーを使用不可能にすることはできません。PassTicket は、有効期間が約 10 分のシステム生成パスワードです。生成される PassTicket は、DES 暗号化アルゴリズム、ユーザー ID、アプリケーション ID、時刻と日付のスタンプ、および秘密鍵に基づいています。この秘密鍵は 64 ビットの数値 (16 個の 16 進文字) で、これは、使用するセキュリティ・ソフトウェアに対して定義されている必要があります。さらなるセキュリティのために、z/OS セキュリティ・ソフトウェアは 1 回限りのパスワードとして、デフォルトで PassTicket を処理します。

PassTicket の使用法を理解しやすいように、以下で RSE のセキュリティ・プロセスを簡単に説明します。

1. クライアントは、ホスト・ポート 4035、RSE デーモンに接続します。
2. RSE デーモンはクライアントが提示した資格情報を使用して、クライアントを認証します。
3. RSE デーモンは、固有のクライアント ID と RSE サーバー・スレッドを作成します。
4. RSE サーバーは PassTicket を生成し、その PassTicket をパスワードとして使用して、クライアント用のセキュリティ環境を作成します。
5. クライアントは、RSE デーモンが返したホスト・ポートに接続します。
6. RSE サーバーはクライアント ID を使用して、クライアントの妥当性を検査します。
7. RSE サーバーは新規に作成された PassTicket を、その後のパスワードを必要とするすべてのアクションにパスワードとして使用します。

注: デバッグ・マネージャーでセキュア接続をセットアップするときにも、同様のメカニズムが使用されます。

クライアントの実際のパスワードは、初期認証後は不要になります。SAF 準拠のセキュリティ製品は、PassTicket と正規のパスワードのどちらも評価できるからです。RSE サーバーはパスワードが必要になるたびに、PassTicket を生成して使用します。その結果、PassTicket はクライアントの (一時的に) 有効なパスワードになります。

PassTicket を使用すると、RSE はユーザー固有のセキュリティ環境を任意にセットアップでき、すべてのユーザーの ID とパスワードをテーブルに保管する (これはセキュリティを脅威にさらす可能性があります) 必要がなくなります。また、X.509 証明書など、再使用可能なパスワードを使用しないクライアント認証方式も使用できるようになります。

PassTicket を使用できるようにするには、APPL クラスと PTKTDATA クラスのセキュリティ・プロファイルが必要です。これらのプロファイルはアプリケーション固有のものであるため、現在使用しているシステムのセットアップに影響を及ぼしません。

PassTicket がアプリケーション固有のものであることは、RSE と JES ジョブ・モニターの両方が、同じアプリケーション ID (APPLID) を使用する必要があることを意味します。デフォルトでは、どちらのサーバーも FEKAPPL を APPLID として使用しますが、これは変更でき、そのためには、RSE の場合は rsed.envvars 内で、JES ジョブ・モニターの場合は FEJJCNFG 内で、それぞれ APPLID ディレクティブを使用します。

OMVSAPPL はアプリケーション ID として使用しないでください。これは、大部分の z/OS UNIX アプリケーションの秘密鍵を公開するからです。また、デフォルトの MVS アプリケーション ID (MVS の直後にシステムの SMF ID を続けたもの) も使用しないでください。これは、大部分の MVS アプリケーション (ユーザー・バッチ・ジョブを含む) の秘密鍵を公開するからです。

PassTicket のタイム・スタンプの最小単位は 1 秒です。つまり、同一のアプリケーションが同一のユーザー ID に対して 1 秒以内に生成する PassTicket はすべて同じになるということです。このため、1 回限りのパスワードとして PassTicket を処理する z/OS セキュリティ・ソフトウェアと組み合わせられると、複数の PassTicket が 1 秒以内に要求されることになるので、ログオン中に Developer for System z に問題が発生します。したがって、Developer for System z では、生成された PassTicket の再利用を許可するフラグを PassTicket 定義に設定する必要があります。

重要: PassTicket が正しくセットアップされていないと、クライアントの接続要求は失敗します。

監査ロギング

Developer for System z は、RSE デーモンによって管理されるアクションの監査ロギングをサポートしています。監査ログは、CSV (コンマ区切り値) 形式のテキスト・ファイルとしてデーモン・ログ・ディレクトリーに保管されます。

監査制御

「ホスト構成ガイド」(SC88-5663)の"『_RSE_JAVAOPTS での追加 Java 始動パラメーター』"で説明されているように、rsed.envvars 内の複数のオプションが監査機能に影響します。

- 監査機能は、enable.audit.log オプションによって使用可能/使用不可にされます。
- 監査のデフォルトは、audit.* オプションによって制御されます。
- 監査ログ・ファイルのロケーションは、daemon.log オプションによって制御されます。
- 監査ログの書き込みに使用されるコード・ページは、「ホスト構成ガイド」(SC88-5663) の"『rsed.envvars、RSE 構成ファイル』"で説明しているように、_RSE_HOST_CODEPAGE ディレクティブによって制御されます。

modify switch オペレーター・コマンドを使用して、「ホスト構成ガイド」(SC88-5663)の『オペレーター・コマンド』の説明にあるように、新しい監査ログ・ファイルに手動で切り替えることができます。

監査ログ・ファイルを保持しているファイル・システムのフリー・スペースが少なくなった場合は、コンソールへ警告メッセージが送信されます。このコンソール・メッセージ (FEK103E) は、スペース不足の問題が解決されるまで定期的に繰り返されます。RSE が生成するコンソール・メッセージのリストについては、「ホスト構成ガイド」(SC88-5663)の『コンソール・メッセージ』を参照してください。

監査処理

新しい監査ログ・ファイルは、あらかじめ定められた時間が経過した後、または **modify switch** オペレーター・コマンドが発行されたときに開始されます。古いログ・ファイルは、audit.log.yyyymmdd.hhmmss として保存されます。ここで、yyyymmdd.hhmmss は、そのログが閉じられたときの日付/タイム・スタンプです。ファイルへ割り当てられたシステム日付/タイム・スタンプは、ログ・ファイルの作成を示しています。これら 2 つの日付の組み合わせが、この監査ログ・ファイルの記録期間を示しています。

rsed.envvars の audit.action* ディレクティブを使用すると、監査ログが閉じられたときに RSE によって呼び出されるユーザー出口 (z/OS UNIX シェル・スクリプト、z/OS UNIX REXX、または z/OS UNIX プログラム) を指定できます。このユーザー出口では、監査ログ内のデータを処理できます。

監査ログ・ファイルの許可ビット・マスクは、rsed.envvars の audit.log.mode ディレクティブで変更されていない場合は、640 (-rw-r-----) に設定されています。これは、所有者 (RSE デーモン z/OS UNIX uid) が読み取りアクセス権と書き込みアクセス権を持ち、所有者の (デフォルト) グループが読み取りアクセス権を持つことを意味します。それ以外のアクセスの試みは、すべて拒否されます。ただし、スー

パーユーザー (UID 0) または UNIXPRIV セキュリティー・クラスの SUPERUSER.FILESYS プロファイルに対する十分な権限を持つユーザーが試みた場合を除きます。

監査データ

以下のアクションがログとして記録されます。

- システム・アクセス (接続、切断)
- JES スプール・アクセス (実行依頼、表示、保留、保留解除、キャンセル、ページ)
- データ・セット・アクセス (読み取り、書き込み、作成、削除、名前変更、圧縮、マイグレーション、再呼び出し)
- ファイル・アクセス (読み取り、書き込み、作成、削除、名前変更)
- TSO および z/OS UNIX コマンドの実行

ログに記録された各アクションは、CSV (コンマ区切り値) 形式で (日付/タイム・スタンプ付きで) 保管されます。CSV 形式は、自動化ツールまたはデータ分析ツールで読み取ることができます。以下に例を示します。

```
yyyy/mm/dd hh:mm:ss.sss,userid,action,dataset_name[,returncode]
[,additional_information]]
```

ファイルが開かれる際、データ・セットとメンバーの統計もログとして記録されます。READ アクションの完了を示す行にそれらが付加され、各フィールドは %n 区切り文字で区切られます。以下に例を示します。

```
yyyy/mm/dd hh:mm:ss.sss,userid,action,dataset_name,returncode,create%modify%n...
```

以下の属性が、以下のリストの順でログとして記録されます。

- 作成日時 (mm/dd/yyyy hh:mm)
- 最終変更日時 (mm/dd/yyyy hh:mm:ss)
- 最終アクセス日時 (mm/dd/yyyy hh:mm:ss)
- レコード・フォーマット (RECFM)
- SCLM 改訂標識 (N = 改訂番号が設定されている、D = 改訂番号が設定されていない)
- SCLM 改訂番号
- 「Bad Hex」文字が含まれている (Y = Yes、N = No)

注: これらの「Bad Hex」文字は、Developer for System z マッピング・サービスが必要になります。その理由は、これらの文字は、コード・ページの不一致によりクライアントとの送受信ができないからです。

- 論理レコード長 (LRECL)
- ファイル・サイズ
- 将来の利用のために予約済み
- 将来の利用のために予約済み
- ユーザー ID
- このデータ・セットまたはメンバーに対するロック (エンキュー) 所有者

- CR (キャリッジ・リターン)、LF (ライン・フィード)、および NL (復帰改行) ホスト・コード・ポイント、ならびにそれらの置換文字 (クライアント・バージョン 8.0.3 以降を利用時のみ使用可能)

JES セキュリティー

Developer for System z では、クライアントは JES ジョブ・モニターを通じて JES スプールにアクセスできます。サーバーは基本的なアクセス制限を行い、この制限は、ご使用のセキュリティー製品の標準スプール・ファイル保護フィーチャーによって拡張できます。スプール・ファイルに対するオペレーター・アクション (「保留」、「保留解除」、「キャンセル」、および「ページ」) は EMCS コンソールを通じて行われ、このコンソールについて、条件付きの許可をセットアップする必要があります。

ジョブに対するアクション - ターゲットの制限

JES ジョブ・モニターは、Developer for System z ユーザーに JES スプールへのフル・オペレーター・アクセスを提供するわけではありません。保留、保留解除、キャンセル、ページの各コマンドのみが使用可能で、しかも、デフォルトでは、そのユーザーが所有しているスプール・ファイルの場合だけです。コマンドは、クライアント・メニュー構造で該当するオプションを選択することによって発行されます (コマンド・プロンプトはありません)。コマンドのスコープは、セキュリティー・プロファイルを使用してコマンドの使用対象となるジョブを定義することにより、広げることができます。

SDSF の **SJ** アクション文字と同じように、JES ジョブ・モニターは「JCL の表示」コマンドもサポートしています。このコマンドは、選択されたジョブ出力を作成した JCL を取り出して、エディター・ウィンドウに表示します。JES ジョブ・モニターは JES から JCL を取り出せるので、元の JCL メンバーを容易に見つけることができない状況で役立ちます。

表 1. JES ジョブ・モニターのコンソール・コマンド

アクション	JES2	JES3
保留	\$Hx(jobid) x = {J、S、または T}	*F,J=jobid,H
保留解除	\$Ax(jobid) x = {J、S、または T}	*F,J=jobid,R
キャンセル	\$Cx(jobid) x = {J、S、または T}	*F,J=jobid,C
ページ	\$Cx(jobid),P x = {J、S、または T}	*F,J=jobid,C
JCL の表示	適用外	適用外

表 1 にリストした使用可能な JES コマンドは、デフォルトでは、そのユーザーが所有しているジョブだけに制限されます。これは、「ホスト構成ガイド」

(SC88-5663) の"『FEJJC�FG、JES ジョブ・モニター構成ファイル』"で説明されているように、LIMIT_COMMANDS ディレクティブを使用して変更できます。

表 2. LIMIT_COMMANDS コマンドの許可のマトリックス

	ジョブ所有者	
LIMIT_COMMANDS	ユーザー	その他
USERID (デフォルト)	許可される	許可されない
LIMITED	許可される	セキュリティ・プロファイルによって明示的に許可された場合にのみ許可される
NOLIMIT	許可される	セキュリティ・プロファイルによって許可された場合、または JESSPOOL クラスがアクティブでない場合は許可される

JES は、JESSPOOL クラスを使用して SYSIN/SYSOUT データ・セットを保護します。SDSF と同じように、JES ジョブ・モニターは JESSPOOL クラスの用途を拡張し、ジョブ・リソースの保護も行います。

LIMIT_COMMANDS が USERID でない場合、JES ジョブ・モニターは、次の表に示すように、JESSPOOL クラス内の関連するプロファイルに対する権限を照会します。

表 3. 拡張 JESSPOOL プロファイル

コマンド	JESSPOOL プロファイル	必要なアクセス権
保留	nodeid.userid.jobname.jobid	ALTER
保留解除	nodeid.userid.jobname.jobid	ALTER
キャンセル	nodeid.userid.jobname.jobid	ALTER
パージ	nodeid.userid.jobname.jobid	ALTER
JCL の表示	nodeid.userid.jobname.jobid.JCL	READ

前記の表で、以下のように置き換えてください。

nodeid	ターゲット JES サブシステムの NJE ノード ID
userid	ジョブ所有者のローカル・ユーザー ID
jobname	ジョブの名前
jobid	JES ジョブ ID

JESSPOOL クラスがアクティブでない場合、LIMIT_COMMANDS の LIMITED および NOLIMIT 値の動作は、「ホスト構成ガイド」(SC88-5663) の"『FEJJC�FG、JES ジョブ・モニター構成ファイル』"の表『LIMIT_COMMANDS コマンド許可マトリックス』"の説明のように異なります。JESSPOOL がアクティブの場合、動作は同じです。クラスは、デフォルトでは、プロファイルが定義されていなければ許可を拒否するからです。

アクション、ジョブに対する - 実行の制限

許可されたターゲットを指定した後の、JES スプール・コマンド・セキュリティの第 2 フェーズには、オペレーター・コマンドを実際に実行するために必要な許可が含まれます。この実行許可は、z/OS および JES のセキュリティ検査によって行われます。

「JCL の表示」は、他の JES ジョブ・モニター・コマンド（「保留」、「保留解除」、「キャンセル」、および「パージ」）のようなオペレーター・コマンドでないため、次のリストに挙げる制限が（それ以上のセキュリティ検査が存在しないので）適用されないことに注意してください。

JES ジョブ・モニターは、ユーザーが要求したすべての JES オペレーター・コマンドを、拡張 MCS (EMCS) コンソールを通じて発行します。このコンソールの名前は、「ホスト構成ガイド」(SC88-5663) の『FEJJC�FG、JES ジョブ・モニター構成ファイル』の説明にあるように、CONSOLE_NAME ディレクティブによって制御されます。

JES ジョブ・モニターを使用すると、LIMIT_CONSOLE ディレクティブを使用して EMCS コンソールに付与する権限の程度を定義できます。この説明は、「ホスト構成ガイド」(SC88-5663) の『FEJJC�FG - JES ジョブ・モニター構成ファイル』に説明されています。

表 4. LIMIT_CONSOLE コンソールの権限マトリックス

LIMIT_CONSOLE	OPERCMDS クラス内のアクティブなプロファイル	OPERCMDS クラス内のアクティブでないプロファイル
LIMITED (デフォルト)	許可される (セキュリティ・プロファイルで許可されている場合)	許可されない
NOLIMIT	許可される (セキュリティ・プロファイルで許可されている場合)	許可される

このセットアップでは、セキュリティ管理者は OPERCMDs クラスおよび CONSOLE クラスを使用して、詳細なコマンド実行権限を定義できます。

- EMCS コンソールを使用するユーザーは、OPERCMDs クラス内の MVS.MCSOPER.console-name プロファイルに対する READ 権限を (最低限) 持っている必要があります。プロファイルが定義されていない場合、システムは権限要求を認可することに注意してください。
- JES オペレーター・コマンドを実行するユーザーは、OPERCMDs クラス内の JES%.** (または、それ以上の詳細な) プロファイルに対する十分な権限を持っている必要があります。LIMIT_CONSOLE=LIMITED が FEJJC�FG に定義されているときに、プロファイルが定義されていないか、OPERCMDs クラスがアクティブでない場合、JES はコマンドの実行に失敗します。
- セキュリティ管理者は、**PERMIT** 定義で WHEN(CONSOLE(JMON)) を指定することにより、ユーザーがオペレーター・コマンドを実行するときに、JES ジョブ・モニターの使用を必須とすることができます。このセットアップを機能させるには、CONSOLE クラスをアクティブにする必要があります。CONSOLE クラスがアク

タイプであれば十分であることに注意してください。EMCS コンソールの有無について、プロファイルが検査されることはありません。

TSO セッションから JMON コンソールを作成することによって JES ジョブ・モニター・サーバーの ID を装うことは、セキュリティ・ソフトウェアによって防止されます。コンソールを作成できる場合でも、(JES ジョブ・モニターと TSO とでは) 入り口点が異なります。この資料で説明されているとおりにセキュリティがセットアップされており、ユーザーが他の手段によって JES コマンドに対する権限を持っていない場合は、そのコンソールから発行された JES コマンドはセキュリティ検査で不合格になります。

JES ジョブ・モニターは、コマンドを実行する必要があるとき、コンソール名が既に使用されていると、コンソールを作成できないことに注意してください。これを防止するために、システム・プログラマーは JES ジョブ・モニター構成ファイル内で GEN_CONSOLE_NAME=ON ディレクティブを設定でき、セキュリティ管理者は TSO ユーザーがコンソールを作成しないようにセキュリティ・プロファイルを定義できます。以下のサンプル RACF コマンドは、(許可されたユーザーを除いて) 誰も TSO または SDSF コンソールを作成できないようにします。

- RDEFINE TSOAUTH CONSOLE UACC(NONE)
- PERMIT CONSOLE CLASS(TSOAUTH) ACCESS(READ) ID(#userid)
- RDEFINE SDSF ISFCMD.ODSP.ULOG.* UACC(NONE)
- PERMIT ISFCMD.ODSP.ULOG.* CLASS(SDSF) ACCESS(READ) ID(#userid)

注: これらのオペレーター・コマンドの許可がない場合でも、ユーザーは、これらのリソースを保護できるプロファイル (JESINPUT、JESJOBS、および JESSPOOL クラス内のプロファイルなど) に対して十分な権限を持っていれば、JES ジョブ・モニターを通じてジョブを実行依頼し、ジョブ出力を読み取ることができます。

オペレーター・コマンド保護の詳細については、「*Security Server RACF セキュリティ管理者のガイド*」(SA88-8613) を参照してください。

スプール・ファイルへのアクセス

JES ジョブ・モニターは、デフォルトでは、すべてのスプール・ファイルへの参照アクセスを許可します。これは、「*ホスト構成ガイド*」(SC88-5663) の

『FEJJCNFG、JES ジョブ・モニター構成ファイル』"で説明されているように、LIMIT_VIEW ディレクティブを使用して変更できます。

表 5. LIMIT_VIEW のブラウズ権限のマトリックス

LIMIT_VIEW	ジョブ所有者	
	ユーザー	その他
USERID	許可される	許可されない
NOLIMIT (デフォルト)	許可される	セキュリティ・プロファイルによって許可された場合、または JESSPOOL クラスがアクティブでない場合は許可される

ユーザーを JES スプール上のそのユーザー自身のジョブだけに制限するには、JES ジョブ・モニター構成ファイル FEJJCNFG で "LIMIT_VIEW=USERID" ステートメントを定義します。すべてのジョブではありませんが、より広範囲のジョブへのアクセスをユーザーが必要とする場合は、使用しているセキュリティ製品の標準スプール・ファイル保護フィーチャー、例えば、JESSPOOL クラスなどを使用します。

さらに保護を定義するときは、JES ジョブ・モニターがスプールへのアクセスに SAPI (SYSOUT アプリケーション・プログラム・インターフェース) を使用することに留意してください。これは、ユーザーが単にブラウズ機能を使用するだけでも、最低限、スプール・ファイルに対する UPDATE アクセス権が必要になることを意味します。この必要条件は、z/OS 1.7 (JES3 の場合は z/OS 1.8) 以上を実行している場合には適用されません。この場合、ブラウズ機能を使用するには、READ 権限で十分です。

JES スプール・ファイルの保護の詳細については、「*Security Server RACF セキュリティ管理者のガイド*」(SA88-8613) を参照してください。

SSL/TLS 暗号化通信

RSE を使用した外部 (クライアント/ホスト) 通信を、SSL (Secure Sockets Layer) またはトランスポート層セキュリティ (TLS) を使用して暗号化できます。このフィーチャーは、デフォルトでは使用不可に設定され、`ssl.properties` 内の設定によって制御されます。「*ホスト構成ガイド*」(SC88-5663) の「『(オプション) `ssl.properties`、RSE SSL 暗号』」を参照してください。

RSE デーモンおよび RSE サーバーは、両者間のアーキテクチャーの違いから、証明書の保管に関して異なるメカニズムをサポートしています。これは、RSE デーモンと RSE サーバーの両方に SSL 定義と証明書が必要であることを意味しています。RSE デーモンと RSE サーバーが同じ証明書管理方式を使用する場合は、共用証明書を使用できます。

表 6. SSL 証明書の保管メカニズム

証明書ストレージ	作成者および管理者	RSE デーモン	RSE サーバー
鍵リング	SAF 準拠のセキュリティ製品	サポート	サポート
鍵データベース	z/OS UNIX の gskkyman	サポート	/
鍵ストア	Java の keytool	/	サポート

注: SAF 準拠の鍵リングは、証明書の管理に推奨される方式です。

SAF 準拠の鍵リングでは、証明書の秘密鍵が、セキュリティ・データベースに保管されるか、または System z 暗号化ハードウェアとのインターフェースである ICSF (Integrated Cryptographic Service Facility) を使用して保管されます。

ICSF は、非 ICSF の秘密鍵管理よりもセキュアなソリューションであるため、デジタル証明書に関連する秘密鍵の保管には ICSF の使用をお勧めします。ICSF では、確実に、秘密鍵が ICSF マスター・キーで暗号化され、かつその秘密鍵へのアクセスが CSFKEYS および CSFSERV セキュリティ・クラスの一般リソースによって制

御されます。さらに、ICSF はハードウェア Cryptographic Coprocessor (暗号化コプロセッサ) を使用するため、操作上のパフォーマンスが向上します。ICSF の詳細と、暗号鍵およびサービスの使用者を制御する方法については、「*Cryptographic Services ICSF Administrator's Guide*」(SA22-7521) を参照してください。

RSE デーモンは、System SSL の機能を使用して SSL 暗号化通信を管理します。これは SYS1.SIEALNKE が、ご使用のセキュリティー・ソフトウェアによってプログラム制御されることが必要で、LINKLIST または rsed.envvars 内の STEPLIB ディレクティブを介して RSE から使用可能でなければならないことを意味しています。

RSE デーモンまたは RSE サーバーに SAF 準拠の鍵リングが使用されている場合は、RSE ユーザー ID (以下のサンプル・コマンドでは stcrse) に、鍵リングおよび関連する証明書にアクセスするための権限が必要です。

- RDEFINE FACILITY IRR.DIGTCERT.LIST UACC(NONE)
- RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
- PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY) ACCESS(READ) ID(stcrse)
- PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(READ) ID(stcrse)
- SETROPTS RACLIST(FACILITY) REFRESH

rsed.envvars の _RSE_JAVAOPTS ディレクティブの DSTORE_SSL_ALGORITHM 変数によって、SSL とその後継の TLS を暗号化方式として選択できます。詳しくは、「*ホスト構成ガイド*」(SC88-5663) の『_RSE_JAVAOPTS での追加 Java 始動パラメータの定義』を参照してください。

Developer for System z 用に SSL をアクティブにする方法については、205 ページの『第 13 章 SSL および X.509 認証のセットアップ』を参照してください。

統合デバッガーでの暗号化通信

オプションのデバッグ・マネージャーを使用した外部 (クライアント/ホスト) 通信も、SSL または TLS で暗号化できます。この方式で暗号化するには、デバッグ・マネージャーで外部通信に使用されるポート (デフォルトでは 5335) の AT-TLS ポリシーを作成します。35 ページの図 9 にサンプル・ポリシーを示してあります。

AT-TLS (Application Transparent TLS) のセットアップの詳細については、「*Communications Server IP 構成ガイド*」(SC88-8926) を参照してください。

```

| TTLSRule                                RDz_Debug_Manager
| {
|   LocalPortRange                        5335
|   Direction                            Inbound
|   TLSGroupActionRef                    grp_Production
|   TTLSEnvironmentActionRef             RDz_Debug_Manager
| }
| TTLSEnvironmentAction                    RDz_Debug_Manager
| {
|   HandshakeRole Server
|   TLSKeyRingParms
|   {
|     Keyring dbgmgr.racf                # Keyring must be owned by the Debug Manager
|   }
| }
| TLSGroupAction                          grp_Production
| {
|   TTLSEnabled                          On
|   Trace                                2
| }

```

図 9. デバッグ・マネージャーの AT-TLS ポリシー

クライアント認証、X.509 証明書を使用した

RSE デーモンは、X.509 証明書で自分自身を認証するユーザーをサポートしています。この機能の場合、SSL 暗号化通信の使用は前提条件です。この機能は、SSL で使用される証明書でのホスト認証を拡張したものです。

RSE デーモンは、クライアント証明書の妥当性を検査することによって、クライアント認証プロセスを開始します。検査される重要な点は、証明書が有効である日付、および、証明書の署名に使用された認証局 (CA) の信頼性などです。オプションとして、(サード・パーティーの) 証明書失効リスト (CRL) を調べることもできます。

RSE デーモンが証明書の妥当性を検査した後、認証のために証明書が処理されます。証明書は、rsed.envvars ディレクティブの `enable.certificate.mapping` が `false` に設定されている場合を除き、使用しているセキュリティー製品へ渡され、その時点で RSE デーモンは認証を行います。

認証プロセスが成功した場合、そのユーザー ID をこのセッションに使用することが決定され、その後、RSE デーモンはユーザー ID をテストし、RSE デーモンが稼働しているホスト・システム上で使用可能であるかどうかを確認します。

最終検査 (X.509 証明書だけでなく、すべての認証メカニズムで行われます) では、ユーザー ID が Developer for System z の使用を許可されていることが検証されます。

TCP/IP が使用する SSL セキュリティー区分に詳しくれば、上記の検証ステップの組み合わせが「レベル 3 クライアント認証」仕様 (使用可能な最高レベル) に相当することがわかるはずです。

認証局 (CA) の妥当性検査

証明書の妥当性検査の一環として、証明書にユーザーの信頼する証明局 (CA) の署名があるかどうかを検査されます。それを行うために、RSE デーモンは CA を識別する証明書にアクセスできなければなりません。

SSL 接続に **gskkyman** 鍵データベースを使用している場合は、鍵データベースに CA 証明書を追加する必要があります。

SAF 鍵リングを使用している場合は (これが推奨される方式です)、次のサンプル RACF コマンドに示すように、CA 証明書をセキュリティー・データベースに CERTAUTH 証明書として、TRUST または HIGHTRUST 属性付きで追加する必要があります。

- RACDCERT CERTAUTH ADD(dsn) HIGHTRUST WITHLABEL('label')

ほとんどのセキュリティー製品では、すでにそのデータベースの中に、既知の CA の証明書が NOTRUST 状況付きで入っていることに注意してください。既存の CA 証明書をリストし、割り当てられたラベルに基づいて、その 1 つに信頼できる証明書としてマークを付けるには、次のサンプルの RACF コマンドを使用します。

- RACDCERT CERTAUTH LIST
- RACDCERT CERTAUTH ALTER(LABEL('HighTrust CA')) HIGHTRUST

注: 証明書内の HostIdMappings 拡張に基づいて RACF でユーザーを認証する場合は、HIGHTRUST 状況が必要です。詳しくは、37 ページの『使用しているセキュリティー・ソフトウェアによる認証』を参照してください。

CA 証明書をセキュリティー・データベースに追加した後、このサンプルの RACF コマンドに示すように、CA 証明書を RSE 鍵リングに接続する必要があります。

- RACDCERT ID(stcrse) CONNECT(CERTAUTH LABEL('HighTrust CA')
RING(rdzssl.racf))

RACDCERT コマンドの詳細については、「*Security Server RACF コマンド言語解説書*」(SA88-8617) を参照してください。

重要: セキュリティー・ソフトウェアではなく RSE デーモンでユーザーを認証する場合は、SAF 鍵リングまたは **gskkyman** 鍵データベース内で TRUST 状況と HIGHTRUST 状況の CA が混在しないように注意する必要があります。RSE デーモンは、この 2 つを区別できないので、TRUST 状況の CA によって署名された証明書がユーザー ID 認証用に有効になります。

(オプション) 証明書失効リスト (CRL) に対する照会

必要であれば RSE デーモンに指示して、1 つ以上の証明書失効リスト (CRL) を検査させ、妥当性検査プロセスに追加のセキュリティーを付加できます。これは、CRL に関連した環境変数を `rsed.envvars` に追加することによって行います。

- GSK_CRL_SECURITY_LEVEL
- GSK_LDAP_SERVER
- GSK_LDAP_PORT
- GSK_LDAP_USER
- GSK_LDAP_PASSWORD

これらの環境変数、および z/OS System SSL によって使用されるその他の環境変数の詳細については、「*Cryptographic Services System SSL (Secure Sockets Layer) プログラミング*」(SD88-6252) を参照してください。

注: 他の z/OS System SSL 環境変数 (GSK_*) を rsed.envvars で指定するときは、それらの環境変数によって、RSE デーモンによる SSL 接続と証明書認証の処理方法が変更される可能性があるので注意してください。

使用しているセキュリティ・ソフトウェアによる認証

RACF は、いくつかの検査を行って証明書を認証し、関連するユーザー ID を返します。他のセキュリティ製品では、これが異なる方法で行われる場合があることに注意してください。認証 (照会モード) を行うために使用される initACEE 機能の詳細については、使用しているセキュリティ製品の資料を参照してください。

1. RACF は、証明書が DIGTCERT クラス内で定義されているかどうかを検査します。定義されている場合、RACF は、その証明書が RACF データベースに追加されたときにその証明書に関連付けられていたユーザー ID を返します。

証明書を RACF に対して定義するには、次の例に示すような RACDCERT コマンドを使用します。

```
RACDCERT ID(userid) ADD(dsn) TRUST WITHLABEL('label')
```

2. 証明書が定義されていない場合、RACF は DIGTNMAP クラスまたは DIGTCRIT クラス内に一致する証明書名フィルターが定義されているかどうかを調べます。定義されていれば、最もよく一致するフィルターに関連したユーザー ID を返します。

注: Developer for System z が使用する証明書には、名前フィルターを使用しないことをお勧めします。それらのフィルターによって、すべての証明書が単一のユーザー ID にマップされてしまうからです。その結果、すべての Developer for System z ユーザーが同じユーザー ID でログオンすることになります。

3. 一致する名前フィルターがない場合、RACF は HostIdMappings 証明書拡張を見つけ、埋め込まれたユーザー ID とホスト名のペアを抽出します。検出され、妥当性が確認された場合、RACF は HostIdMappings 拡張の中で定義されているユーザー ID を返します。

ユーザー ID とホスト名のペアは、以下のすべての条件が真である場合に有効です。

- この証明書に署名するために使用された CA 証明書に、DIGTCERT クラス内で HIGHTRUST のマークが付いている。
- 拡張内に保管されているユーザー ID の長さが妥当 (1 から 8 文字まで) である。
- RSE デーモンへ割り当てられたユーザー ID が、SERVAUTH クラス内の IRR.HOST.hostname プロファイルに対して (最低でも) READ 権限を持っている。ここで、hostname は拡張内に保管されているホスト名です。これは通常、CDFMVS08.RALEIGH.IBM.COM などのドメイン・ネームです。

ASN.1 構文での HostIdMappings 拡張の定義は、以下のとおりです。

```
id-ce-hostIdMappings OBJECT IDENTIFIER ::= {1 3 18 0 2 18 1}
HostIdMappings ::= SET OF HostIdMapping
HostIdMapping ::= SEQUENCE{
    hostName          IMPLICIT[1] IA5String,
    subjectId         IMPLICIT[2] IA5String,
    proofOfIdPossession IdProof OPTIONAL
}
```

```

IdProof ::= SEQUENCE{
    secret      OCTET STRING,
    encryptionAlgorithm OBJECT IDENTIFIER
}

```

注: HostIdMappings 拡張は、ターゲット・ユーザー ID が、その HostIdMappings 拡張を含んでいる証明書の有効期間の開始後に作成されている場合、受け入れられません。したがって、特に HostIdMappings 拡張を持つ証明書用にユーザー ID を作成する場合は、必ず、証明書要求を実行依頼する前にユーザー ID を作成しておいてください。

X.509 証明書、RACF によるその管理方法、および証明書名フィルターの定義方法について詳しくは、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613)を参照してください。 **RACDCERT** コマンドの詳細については、「*Security Server RACF コマンド言語解説書*」(SA88-8617)を参照してください。

RSE デーモンによる認証

Developer for System z では、ご使用のセキュリティー製品に依存せずに基本的な X.509 証明書認証を行うことができます。RSE デーモンによって行われる認証では、ユーザー ID とホスト名が証明書拡張内で定義されている必要があり、しかも、rsed.envvars 内の enable.certificate.mapping ディレクティブが FALSE に設定されている場合にのみ、認証がアクティブになります。

この機能は、使用しているセキュリティー製品が X.509 証明書に基づいたユーザーの認証をサポートしていない場合、または、使用しているセキュリティー製品によって行われるテストに証明書が合格できない場合 (例えば、証明書の HostIdMappings 拡張の ID に誤りがあり、DIGTCERT 内に名前フィルターや定義がない場合) に使用するためのものです。

クライアントはユーザーに対し、使用すべき拡張 ID (OID) を照会し、その ID は、デフォルトでは HostIdMappings OID の {1 3 18 0 2 18 1} です。

RSE デーモンは HostIdMappings 拡張のフォーマットを使用して、そこからユーザー ID とホスト名を抽出します。このフォーマットについては、37 ページの『使用しているセキュリティー・ソフトウェアによる認証』に説明があります。

ユーザー ID とホスト名のペアは、以下のすべての条件が真である場合に有効です。

- 拡張内に保管されているユーザー ID の長さが妥当 (1 から 8 文字まで) である。
- RSE デーモンへ割り当てられたユーザー ID が、SERVAUTH クラス内の IRR.HOST.hostname プロファイルに対して (最低でも) READ 権限を持っている。ここで、hostname は拡張内に保管されているホスト名です。これは通常、CDFMVS08.RALEIGH.IBM.COM などのドメイン・ネームです。

重要: RSE デーモンに既知のすべての CA を「高度に信頼できる」ものと保証するのは、セキュリティ管理者の責任です。RSE デーモンはクライアント証明書の署名者が「高度に信頼できる」か単なる「信頼できる」かを検査できないからです。アクセス可能な CA 証明書の詳細については、35 ページの『認証局 (CA) の妥当性検査』を参照してください。

Port Of Entry (POE) 検査

Developer for System z は Port Of Entry (POE) 検査をサポートしています。これにより、ホストは信頼できる TCP/IP アドレスにのみアクセスできます。このフィーチャーは、デフォルトでは使用不可に設定され、以下のサンプル RACF コマンドに示すように、BPX.POE セキュリティー・プロファイルの定義を必要とします。

- RDEFINE FACILITY BPX.POE UACC(NONE)
- PERMIT BPX.POE CLASS(FACILITY) ACCESS(READ) ID(STCRSE)
- SETROPTS RACLIST(FACILITY) REFRESH

注:

- 「ホスト構成ガイド」(SC88-5663)の"『_RSE_JAVAOPTS での追加 Java 始動パラメーター』"の説明にあるように、rsed.envvars で「enable.port.of.entry=true」オプションをコメント解除することにより、POE を使用するよう RSE を構成する必要があります。
- RSE ユーザー ID STCRSE は、このプロファイルが定義されておらず、rsed.envvars 内で POE 検査が使用可能にされている場合、UID(0) を必要とします。
- BPX.POE を定義すると、POE 検査をサポートしている他の TCP/IP アプリケーション (INETD など) に影響が及びます。
- POE 検査の強度を完全に活用するには、SERVAUTH クラス内でセキュリティ・ゾーン (IP アドレス範囲である EZB.NETACCESS.** プロファイル) をセットアップしてください。

POE 検査を使用したネットワーク・アクセス制御の詳細については、「*Communications Server IP 構成ガイド*」(SC88-8926) を参照してください。

クライアント関数の変更

Developer for System z クライアント・バージョン 8.5.1 以降は、SAF セキュリティー・プロファイルのアクセス権限をチェックでき、結果に基づいてユーザーに対し、関連する関数の有効化または無効化を行います。

Developer for System z は、ユーザーに対してどのオプションを有効または無効にするかを決定するため、40 ページの表 7 にリストされているプロファイルに対してアクセス権限を検証します。

表 7. クライアント関数の変更のための SAF 情報

FACILITY プロファイル	固定長	必要なアクセス権	結果
FEK.USR.OFF.REMOTECOPY.MVS.sysname	27	READ	クライアントは、MVS データ・セットのコピーおよび関連した関数を無効にします。

注: プロファイルへのアクセス許可がユーザーに付与されているかどうか判断できないことをセキュリティー・ソフトウェアが示した場合、Developer for System z そのユーザーにはアクセス許可が付与されていないと仮定します。この一例として、プロファイルが定義されていない場合があります。

sysname の値は、ターゲット・システムのシステム名と一致します。

「固定長」列は、関連するセキュリティー・プロファイルの固定部分の長さについて説明しています。

デフォルトでは、Developer for System z は、FEK.* プロファイルが FACILITY セキュリティー・クラスに存在していると想定します。FACILITY クラス内のプロファイルには 39 文字までの文字数制限があることに注意してください。プロファイルの固定部分 (FEK.USR.<key>) の長さと、プロファイルのサイト固有部分(sysname) の長さの合計がこの制限数を超過する場合は、プロファイルを別のクラス内に置き、代わりにこのクラスを使用するように Developer for System z に指示してください。これを行うには、rsed.envvars にある _RSE_FEK_SAF_CLASS をコメント解除して、適切なクラス名を指定します。

以下のサンプルのセキュリティー定義は REMOTECOPY.MVS 操作を、RESTRICT グループ内のユーザーを除く CDFMVS08 上のすべてのユーザーで許可します。

```
RDEFINE FACILITY (FEK.USR.OFF.REMOTECOPY.MVS.CDFMVS08) -
  UACC(NONE) DATA('RATIONAL DEVELOPER FOR SYSTEM Z - CLIENT CONTROL')
PERMIT FEK.USR.OFF.REMOTECOPY.MVS.CDFMVS08 CLASS(FACILITY) -
  ID(RESTRICT) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

OFF.REMOTECOPY.MVS

ユーザーが FEK.USR.OFF.REMOTECOPY.MVS.sysname プロファイルの READ アクセス権限を持っている場合、Developer for System z クライアント・バージョン 8.5.1 以降では、MVS データ・セットに対するドラッグ、コピー、別名保存およびオフライン作業ができなくなります。結果として、ユーザーはシステム上のデータ・セットへアクセスできますが、ワークステーション上にデータ・セットのローカル・コピーを作成することはできません。これは、ローカル・ワークステーションの紛失や盗難があった場合に、機密情報の漏えいを防ぐのに役立ちます。

クライアントへのプッシュの開発者グループ

Developer for System z クライアント・バージョン 8.0.1 以上は、接続時にホストからクライアントの構成ファイルとアップグレード情報を取り出すことができるので、すべてのクライアントの設定が共通になり、最新のものになります。

バージョン 8.0.3 以降、クライアント管理者は、異なる開発者グループのニーズに適合するように、複数のクライアント構成のセットと複数のクライアント更新シナリオを作成できるようになりました。これにより、ユーザーは、LDAP グループのメンバーシップやセキュリティ・プロファイルに対する許可などの基準に基づいてカスタマイズされたセットアップを受け取れるようになります。

セキュリティ・データベースの定義を選択手段として使用する場合 (pushtoclient.properties のディレクティブに SAF 値が指定されている場合)、Developer for System z は 表 8 にリストされているプロファイルへのアクセス権限を検証して、ユーザーが所属している開発者グループを判別したり、ユーザーが更新の拒否を許可されているかどうかを判別します。

表 8. クライアントへのプッシュの SAF 情報

FACILITY プロファイル	固定長	必要なアクセス権	結果
FEK.PTC.CONFIG.ENABLED. sysname.devgroup	23	READ	クライアントは指定されたグループ用の構成の更新を受諾する
FEK.PTC.PRODUCT. ENABLED.sysname.devgroup	24	READ	クライアントは指定されたグループ用の製品の更新を受諾する
FEK.PTC.REJECT.CONFIG. UPDATES.sysname	30	READ	ユーザーは構成の更新を拒否できる
FEK.PTC.REJECT.PRODUCT. UPDATES.sysname	31	READ	ユーザーは製品の更新を拒否できる

注: プロファイルへのアクセス許可がユーザーに付与されているかどうかは判断できないことをセキュリティ・ソフトウェアが示した場合、Developer for System z そのユーザーにはアクセス許可が付与されていないと仮定します。この一例として、プロファイルが定義されていない場合があります。

devgroup 値は、特定の開発者グループに割り当てられたグループ名と一致します。グループ名は、Developer for System z クライアントに表示されることに注意してください。

sysname の値は、ターゲット・システムのシステム名と一致します。

「固定長」列は、関連するセキュリティ・プロファイルの固定部分の長さについて説明しています。

デフォルトでは、Developer for System z は、FEK.* プロファイルが FACILITY セキュリティ・クラスに存在していると想定します。FACILITY クラス内のプロファイ

ルには 39 文字までの文字数制限があることに注意してください。プロファイルの固定部分 (FEK.PTC.<key>) の長さと、プロファイルのサイト固有部分 (sysname または sysname.devgroup) の長さの合計が、この制限数を超過する場合は、プロファイルを別のクラス内に置き、代わりにこのクラスを使用するように Developer for System z に指示してください。これを行うには、rsed.envvars にある _RSE_FEK_SAF_CLASS をコメント解除して、適切なクラス名を指定します。

関連するクライアントへのプッシュ・メタデータを定義および管理するために、クライアント管理者は FEK.PTC.*.ENABLED.* プロファイルのアクセス・リストに記載されている必要があります。これは、グループ・サポートを伴うクライアントへのプッシュを実装するには、(少なくとも) アクセス・リストに記載されたクライアント管理者と一緒に、事前にプロファイルを定義しておく必要がある、ということです。

複数グループのサポートを有効にする方法についての詳細は、「[ホスト構成ガイド](#)」(SC88-5663) の『(オプション) pushtoclient.properties、ホスト・ベースのクライアント制御』を参照してください。クライアントへのプッシュの概念と実装について詳しくは、131 ページの『[第 7 章 クライアントへのプッシュの考慮事項](#)』を参照してください。

デバッグ・セキュリティー

オプションの統合デバッガーを使用すると、読み取り専用メモリーにロードされた CICS トランザクションをデバッグできます。問題プログラム状態 (不許可) のトランザクションのデバッグに使用する場合、統合デバッガーは、デバッグ・セッションの所有ユーザーがその実行を許可されているかを確認します。

Developer for System z は、表 9 にリストされたプロファイルへのアクセスを確認して、デバッグ権限が付与されているかを判別します。

表 9. デバッグ機能のための SAF 情報

FACILITY プロファイル	必要なアクセス権	結果
AQE.AUTHDEBUG.WRITEBUFFER	UPDATE	ユーザーは読み取り専用 CICS トランザクションのデバッグが可能

注: プロファイルへのアクセス許可がユーザーに付与されているかどうか判断できないことをセキュリティー・ソフトウェアが示した場合、Developer for System z そのユーザーにはアクセス許可が付与されていないと仮定します。この一例として、プロファイルが定義されていない場合があります。

以下のサンプルのセキュリティー定義は、AUTHDEBUG.WRITEBUFFER 操作を RDZDEBUG グループ内のすべてのユーザーに許可します。

```
RDEFINE FACILITY (AQE.AUTHDEBUG.WRITEBUFFER) -
  UACC(NONE) DATA('RATIONAL DEVELOPER FOR SYSTEM Z - DEBUG CICS READ-ONLY')
PERMIT AQE.AUTHDEBUG.WRITEBUFFER CLASS(FACILITY) -
  ID(RDZDEBUG) ACCESS(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
```

CICSTS セキュリティー

オプションの統合デバッガーを使用すると CICS トランザクションをデバッグできます。詳しくは、163 ページの『CICS トランザクションのデバッグ』を参照してください。

Developer for System z では、Application Deployment Manager を通じて、開発者が編集可能な CICS リソース定義とそのデフォルト値、および CICS リソース定義の表示を、CICS 管理者が CICS リソース定義 (CRD) サーバーによって制御できます。必要な CICS TS セキュリティー定義の詳細については、151 ページの『第 8 章 CICSTS に関する考慮事項』を参照してください。

CRD リポジトリ

CRD サーバー・リポジトリ VSAM データ・セットは、すべてのデフォルト・リソース定義を保持しています。したがって、更新されないように保護する必要がありますが、開発者は、そこに保管された値の読み取りを許可される必要があります。

CICS トランザクション

Developer for System z は、CICS リソースの定義および照会時に、CRD サーバーが使用する複数のトランザクションを提供します。トランザクションが接続すると、CICS リソース・セキュリティ検査機能 (使用可能な場合) により、ユーザー ID にはそのトランザクション ID を実行する許可が与えられます。

SSL 暗号化通信

Application Deployment Manager クライアントは、CICS TS Web サービスまたは RESTful インターフェースを使用して、CRD サーバーを起動します。この通信への SSL の使用は、CICS TS TCIPSERVICE 定義によって制御されます。詳しくは「*RACF Security Guide for CICS TS*」を参照してください。

各種情報

GATE の処分

アドレス・スペースが RACF に、DATASET クラスのような RACLIST 処理 (メモリー内に保管) されていないリソース・クラスへのアクセスを指示する初回に、RACF は、ユーザーのアドレス・スペースにある関連するすべての総称プロファイルを取り出し、GATE (Generic Anchor Table Entry) と呼ばれるリストに保管します。z/OS 1.12 までは、RACF は総称アンカーをアドレス・スペースごとに 4 つ、独自の ACEE を持つ MVS TCB ごとに 4 つ維持します。4 つすべてを使い果たしたとき、RACF は、新しいものが入ってきたときに、最も長い間参照されていないものを置き換えます。

ユーザーが頻繁に 4 つを超えるデータ・セットの高位修飾子にアクセスする場合、RACF は使用可能なアンカー・スロットを介して新規エンTRIESを順番に使用する必要があるため、RSE スレッド・プール (スレッドを使用して複数のユーザーにユーザー固有の ACEE を提供する) で、GATE の処分が発生する可能性があります。

z/OS 1.12 では、RACF に「SET」コマンドの「GENERICANCHOR」オプションが導入され、これによりテーブルのサイズを大きくすることができます。これはシステム全体に、または各ジョブ名ごとに設定できます。

管理 ACEE

Developer for System z では、pthread_security_np() や __passwd() などの z/OS UNIX カーネル・サービスを使用します。これらは InitACEE セキュリティー・サービスを使用するため、結果として「管理 ACEE」セキュリティ管理ブロックになります。管理 ACEE (Accessor Environment Element) は、セキュリティ製品によってキャッシュされるので、特定の変更 (Developer for System z の外部で行ったパスワードの変更など) はキャッシュの期限が切れるまでセキュリティ製品によって無視されます (期限が切れるまでに数分かかることがあります)。

セキュリティを変更した後は、管理 ACEE キャッシュを更新して、確実に新しいデータが Developer for System z で使用されるようにしてください。

SCLM セキュリティー

SCLM Developer Toolkit サービスは、ビルド、プロモート、およびデプロイ機能に対するオプションのセキュリティ機能を提供します。

SCLM 管理者による機能に対してセキュリティが有効の場合、保護された機能を呼び出し元ユーザー ID または代理ユーザー ID で実行する権限を確認するために、SAF 呼び出しが行われます。

必要な SCLM セキュリティー定義の詳細については、「*SCLM Developer Toolkit 管理者ガイド*」(SC88-5664) を参照してください。

Developer for System z 構成ファイル

セキュリティおよび監査のセットアップに影響するディレクティブは、Developer for System z の複数の構成ファイルに存在します。この章の情報に基づいて、セキュリティ管理者およびシステム・プログラマーは、以下のディレクティブの設定内容を決定することができます。

JES ジョブ・モニター - FEJJCNFG

- LIMIT_COMMANDS={USERID | LIMITED | NOLIMIT}

どのジョブに対してアクションを実行できるかを定義します (ブラウズと実行依頼は除く)。詳しくは、29 ページの『ジョブに対するアクション - ターゲットの制限』を参照してください。

- LIMIT_CONSOLE={LIMITED | NOLIMIT}

アクションの実行に使用される EMCS コンソールの権限レベルを定義します。詳しくは、29 ページの『ジョブに対するアクション - ターゲットの制限』を参照してください。

- LIMIT_VIEW={USERID | NOLIMIT}

どのスプール・ファイルをブラウズできるかを定義します。詳しくは、32 ページの『スプール・ファイルへのアクセス』を参照してください。

- `LOOPBACK_ONLY={ON | OFF}`

この z/OS システム外から JES ジョブ・モニターにアクセスできるかどうかを定義します。詳しくは、「[ホスト構成ガイド](#)」(SC88-5663) の章『[基本的なカスタマイズ](#)』の『[FEJJCNFG、JES ジョブ・モニター構成ファイル](#)』を参照してください。

- `APPLID={FEKAPPL | *}`

PassTicket の作成と妥当性検査に使用するアプリケーション ID。詳しくは、25 ページの『[PassTicket の使用](#)』を参照してください。

注: 上記の、およびその他の FEJJCNFG ディレクティブの詳細については、「[ホスト構成ガイド](#)」(SC88-5663) の「『[FEJJCNFG、JES ジョブ・モニター構成ファイル](#)』」に説明があります。

RSE - rsed.envvars

- `_RSE_FEK_SAF_CLASS={FACILITY | *}`

FEK.** プロファイルを保持するセキュリティー・クラス。詳しくは、41 ページの『[クライアントへのプッシュの開発者グループ](#)』および 39 ページの『[クライアント関数の変更](#)』を参照してください。

- `(_RSE_JAVAOPTS) -DDENY_PASSWORD_SAVE={true | false}`

ユーザーがホスト・パスワードをクライアント上に保存することを禁止します。詳しくは、「[ホスト構成ガイド](#)」(SC88-5663) の「『[_RSE_JAVAOPTS](#)」での追加 Java 始動パラメーター」を参照してください。

- `(_RSE_JAVAOPTS) -DDSTORE_IDLE_SHUTDOWN_TIMEOUT=value`

活動停止中のクライアントを切断するタイマー。詳しくは、「[ホスト構成ガイド](#)」(SC88-5663) の「『[_RSE_JAVAOPTS](#)」での追加 Java 始動パラメーター」を参照してください。

- `(_RSE_JAVAOPTS) -DAPPLID={FEKAPPL | *}`

PassTicket の作成と妥当性検査に使用するアプリケーション ID。詳しくは、25 ページの『[PassTicket の使用](#)』を参照してください。

- `(_RSE_JAVAOPTS) -Denable.port.of.entry={true | false}`

Port Of Entry 検査を使用可能にします。詳しくは、39 ページの『[Port Of Entry \(POE\) 検査](#)』を参照してください。

- `(_RSE_JAVAOPTS) -DDSTORE_SSL_ALGORITHM={TLSv1.2 | SSL}`

通信の暗号化方式として SSL か TLS を選択します。詳しくは、33 ページの『[SSL/TLS 暗号化通信](#)』を参照してください。

- `(_RSE_JAVAOPTS) -Denable.certificate.mapping={true | false}`

セキュリティー製品を使用して、X.509 証明書でユーザーを認証します。詳しくは、35 ページの『クライアント認証、X.509 証明書を使用した』を参照してください。

- `GSK_CRL_SECURITY_LEVEL={LOW | MEDIUM | HIGH}`

`GSK_LDAP_SERVER=*`
`GSK_LDAP_PORT={389 | *}`
`GSK_LDAP_USER=*`
`GSK_LDAP_PASSWORD=*`

X.509 認証のための追加のセキュリティー検査。詳しくは、36 ページの『(オプション) 証明書失効リスト (CRL) に対する照会』を参照してください。

- `(_RSE_JVAOPTS) -Ddaemon.log={/var/rdz/logs | *}`

監査ログ・ファイルのロケーション。詳しくは、27 ページの『監査ロギング』を参照してください。

- `(_RSE_JVAOPTS) -Daudit.log.mode={RW.R. | * }`

監査ログ・ファイルのファイル・アクセス許可マスク。詳しくは、27 ページの『監査ロギング』を参照してください。

- `(_RSE_JVAOPTS) -Daudit.action=<shell script>`
`(_RSE_JVAOPTS) -Daudit.action.id=<userid>`

監査ログを処理する z/OS UNIX ベースのユーザー出口。詳しくは、27 ページの『監査ロギング』を参照してください。

注: 上記の、およびその他の `rsed.envvars` ディレクティブの詳細については、「[ホスト構成ガイド](#)」(SC88-5663) の「『`rsed.envvars`、RSE 構成ファイル」」に説明があります。

RSE - `ssl.properties`

- `daemon_keydb_file={SAF key ring name | gskkyman key database name}`

RSE デーモン証明書のロケーション。詳しくは、33 ページの『SSL/TLS 暗号化通信』を参照してください。

- `daemon_key_label=certificate label`

RSE デーモン証明書の名前。詳しくは、33 ページの『SSL/TLS 暗号化通信』を参照してください。

- `server_keystore_file={SAF key ring name | Java key store name}`

RSE サーバー証明書のロケーション。詳しくは、33 ページの『SSL/TLS 暗号化通信』を参照してください。

- `server_keystore_label=certificate label`

RSE サーバー証明書の名前。詳しくは、33 ページの『SSL/TLS 暗号化通信』を参照してください。

- `server_keystore_type={JKS | JCECARCFKS | JCECCARCFKS}`

使用される鍵ストアのタイプ (Java 鍵ストアまたは SAF 鍵リング)。詳しくは、33 ページの『SSL/TLS 暗号化通信』を参照してください。

注: 上記の、およびその他の `ssl.properties` ディレクティブの詳細については、「*ホスト構成ガイド*」(SC88-5663) の「『(オプション) `ssl.properties`、RSE SSL 暗号』」に説明があります。

RSE - `pushtoclient.properties`

- `config.enabled={true | false | SAF | LDAP}`
`reject.config.updates={true | false | SAF | LDAP}`

Developer for System z クライアント構成ファイルをホスト・ベースで制御します。詳しくは、131 ページの『第 7 章 クライアントへのプッシュの考慮事項』を参照してください。

- `product.enabled={true | false | SAF | LDAP}`
`reject.product.updates={true | false | SAF | LDAP}`

Developer for System z クライアント製品更新をホスト・ベースで制御します。詳しくは、131 ページの『第 7 章 クライアントへのプッシュの考慮事項』を参照してください。

注: これらに関する詳細と、その他の `pushtoclient.properties` ディレクティブについては、「*ホスト構成ガイド*」(SC88-5663) の「『(オプション) `pushtoclient.properties`、ホスト・ベースのクライアント制御』」を参照してください。

セキュリティー定義

サンプル FEKRACF メンバーをカスタマイズし、実行依頼してください。このメンバーには、Developer for System z 用の基本セキュリティー定義を作成する、サンプルの RACF および z/OS UNIX コマンドが含まれています。

FEKRACF は `FEK.#CUST.JCL` に置かれます。ただし、`FEK.SFEKSAMP(FEKSETUP)` ジョブをカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳しくは、「*IBM Rational Developer for System z ホスト構成ガイド*」の『カスタマイズのセットアップ』を参照してください。

RACF コマンドの詳細については、「*RACF コマンド言語解説書*」(SA88-8617) を参照してください。

注:

- CA ACF2™ for z/OS を使用しているサイトの場合は、CA サポート・サイト (<https://support.ca.com>) で、ご使用の製品のページを参照し、関連する Developer for System z Knowledge Document TEC492389 を確認してください。この Knowledge Document には、Developer for System z を正しく構成するために必要なセキュリティー・コマンドの詳細が記載されています。
- CA Top Secret® for z/OS を使用しているサイトの場合は、CA サポート・サイト (<https://support.ca.com>) で、ご使用の製品のページを参照し、関連する Developer for System z Knowledge Document TEC492091 を確認してください。こ

の Knowledge Document には、Developer for System z を正しく構成するために必要なセキュリティー・コマンドの詳細が記載されています。

以下のセクションでは、必要なステップ、オプションの構成、および可能な代替策について説明します。

要件およびチェックリスト

セキュリティー・セットアップを完了するために、セキュリティー管理者は表 10 にリストされた値を認識しておく必要があります。これらの値は、前のステップである Developer for System z のインストールとカスタマイズで定義されています。

表 10. セキュリティー・セットアップの変動要素

説明	<ul style="list-style-type: none"> デフォルト値 正解の入手先 	値
Developer for System z 製品の 高位修飾子	<ul style="list-style-type: none"> FEK SMP/E インストール 	
Developer for System z カス タマイズ高位修飾子	<ul style="list-style-type: none"> FEK.#CUST FEK.SFEKSAMP (FEKSETUP)、 詳しくは、「<i>IBM Rational Developer for System z</i> ホ スト構成ガイド」の『カス タマイズのセットアップ』 を参照してください。 	
統合デバッガー開始タスク名	<ul style="list-style-type: none"> DBGMGR FEK.#CUST.PROCLIB (DBGMGR)。詳しくは、 「<i>IBM Rational Developer for System z</i> ホスト構成ガ イド」の『PROCLIB の変 更』を参照してください。 	
JES ジョブ・モニター開始タ スク名	<ul style="list-style-type: none"> JMON FEK.#CUST.PROCLIB (JMON)、詳しくは、「<i>IBM Rational Developer for System z</i> ホスト構成ガイ ド」の『PROCLIB の変 更』を参照してください。 	
RSE デーモン開始タスク名	<ul style="list-style-type: none"> RSED FEK.#CUST.PROCLIB (RSED)、詳しくは、「<i>IBM Rational Developer for System z</i> ホスト構成ガイ ド」の『PROCLIB の変 更』を参照してください。 	

表 10. セキュリティー・セットアップの変動要素 (続き)

説明	<ul style="list-style-type: none"> デフォルト値 正解の入手先 	値
アプリケーション ID	<ul style="list-style-type: none"> FEKAPPL /etc/rdz/rsed.envvars、詳しくは、「<i>IBM Rational Developer for System z</i> ホスト構成ガイド」の『_RSE_JAVAOPTS での追加 Java 始動パラメーターの定義』を参照してください。 	

次のリストは、Developer for System z の基本的なセキュリティー・セットアップを完了するために必要なアクションの概要を示したものです。以下の各セクションで説明されているように、これらの要件を満たすために、必要なセキュリティー・レベルに応じてさまざまな方式を使用できます。オプションの Developer for System z サービスのセキュリティー・セットアップについては、前記の各セクションを参照してください。

- 『セキュリティーの設定およびクラスをアクティブにする』
- 50 ページの『Developer for System z ユーザーの OMVS セグメントを定義する』
- 51 ページの『Developer for System z 開始タスクの定義』
- 52 ページの『セキュアな z/OS UNIX サーバーとして RSE を定義する』
- 53 ページの『RSE の MVS プログラム制御ライブラリーを定義する』
- 54 ページの『RSE の PassTicket サポートを定義する』
- 55 ページの『RSE のアプリケーション保護の定義』
- 56 ページの『JES コマンド・セキュリティーを定義する』
- 58 ページの『データ・セット・プロファイルを定義する』
- 62 ページの『RSE の z/OS UNIX プログラム制御ファイルを定義する』
- 63 ページの『セキュリティー設定の検査』

セキュリティーの設定およびクラスをアクティブにする

Developer for System z では、さまざまなセキュリティー・メカニズムを使用して、クライアントにとってセキュアで制御されたホスト・システム環境を確保します。そのためには、以下のサンプル RACF コマンドで示すように、いくつかのクラスとセキュリティー設定をアクティブにする必要があります。

- 現行の設定を表示する
 - SETROPTS LIST
- z/OS UNIX およびデジタル証明書プロファイルのファシリティ・クラスをアクティブにする
 - SETROPTS GENERIC(FACILITY)
 - SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY)

- 開始タスク定義をアクティブにする
 - SETROPTS GENERIC(STARTED)
 - RDEFINE STARTED ** STDATA(USER(=MEMBER) GROUP(STCGROUP) TRACE(YES))
 - SETROPTS CLASSACT(STARTED) RACLIST(STARTED)
- JES ジョブ・モニターのコンソール・セキュリティーをアクティブにする
 - SETROPTS GENERIC(CONSOLE)
 - SETROPTS CLASSACT(CONSOLE) RACLIST(CONSOLE)
- JES ジョブ・モニターのオペレーター・コマンド保護をアクティブにする
 - SETROPTS GENERIC(OPERCMDS)
 - SETROPTS CLASSACT(OPERCMDS) RACLIST(OPERCMDS)
- RSE のアプリケーション保護をアクティブにする
 - SETROPTS GENERIC(APPL)
 - SETROPTS CLASSACT(APPL) RACLIST(APPL)
- RSE の PassTicket を使用したセキュアなサインオンをアクティブにする
 - SETROPTS GENERIC(PTKTDATA)
 - SETROPTS CLASSACT(PTKTDATA) RACLIST(PTKTDATA)
- 信頼されたコードだけを RSE がロードできるように、プログラム制御をアクティブにする
 - RDEFINE PROGRAM ** ADDMEM('SYS1.CMDLIB'//NOPADCHK) UACC(READ)
 - SETROPTS WHEN(PROGRAM)

注: すでに PROGRAM クラス内に * プロファイルがある場合は、** プロファイルを作成しないでください。セキュリティー・ソフトウェアによって使用される検索パスが、分かりにくく、複雑なものになります。その場合は、既存の * 定義と新しい ** 定義をマージする必要があります。** プロファイルを使用してください。これについては、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) に説明があります。

重要: 「WHEN PROGRAM」がアクティブの場合、一部の製品 (FTP など) はプログラムで制御することが必要です。このプログラム制御は、実動システム上でアクティブにする前にテストしてください。

- (オプション) X.509 HostIdMappings および拡張 Port Of Entry (POE) サポートをアクティブにする
 - SETROPTS GENERIC(SERVAUTH)
 - SETROPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH)

Developer for System z ユーザーの OMVS セグメントを定義する

Developer for System z のユーザーごとに、有効なゼロ以外の z/OS UNIX ユーザー ID (UID)、ホーム・ディレクトリー、およびシェル・コマンドを指定する RACF OMVS セグメントまたは同等のものを定義する必要があります。また、ユーザーのデフォルト・グループも、グループ ID を持つ OMVS セグメントを必要とします。

オプションの統合デバッガーを使用するときには、デバッグするアプリケーションをアクティブ化したユーザー ID、およびそのデフォルト・グループにも、有効な RACF OMVS セグメントまたは同等のものがが必要です。

以下のサンプルの RACF コマンドでは、#userid、#user-identifier、#group-name、および #group-identifier の各プレースホルダーを実際の値に置き換えてください。

- ALTUSER #userid
OMVS(UID(#user-identifier) HOME(/u/#userid) PROGRAM(/bin/sh) NOASSIZEMAX)
- ALTGROUP #group-name OMVS(GID(#group-identifier))

Developer for System z 開始タスクの定義

以下のサンプル RACF コマンドは、保護されたユーザー ID (STCDBGM、STCJMON、および STCRSE) とそれらに割り当てられたグループ FEKD、DBGMGR、JMON、および RSED の各開始タスクを作成します。#group-id および #user-id-* プレースホルダーは、有効な OMVS ID に置き換えてください。

- ADDGROUP STCGROUP OMVS(GID(#group-id))
DATA('GROUP WITH OMVS SEGMENT FOR STARTED TASKS')
- ADDUSER STCDBM DFLTGRP(STCGROUP) NOPASSWORD
NAME('RDZ - DEBUG MANAGER')
OMVS(UID(#user-id-debug) HOME(/tmp) PROGRAM(/bin/sh))
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDUSER STCJMON DFLTGRP(STCGROUP) NOPASSWORD NAME('RDZ - JES JOBMONITOR')
OMVS(UID(#user-id-jmon) HOME(/tmp) PROGRAM(/bin/sh))
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDUSER STCRSE DFLTGRP(STCGROUP) NOPASSWORD NAME('RDZ - RSE DAEMON')
OMVS(UID(#user-id-rse) HOME(/tmp) PROGRAM(/bin/sh) ASSIZEMAX(2147483647)
)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- RDEFINE STARTED DBGMGR.* DATA('RDZ - DEBUG MANAGER')
STDATA(USER(STCDBM) GROUP(STCGROUP) TRUSTED(NO))
- RDEFINE STARTED JMON.* DATA('RDZ - JES JOBMONITOR')
STDATA(USER(STCJMON) GROUP(STCGROUP) TRUSTED(NO))
- RDEFINE STARTED RSED.* DATA('RDZ - RSE DAEMON')
STDATA(USER(STCRSE) GROUP(STCGROUP) TRUSTED(NO))
- SETROPTS RACLIST(STARTED) REFRESH

注:

- NOPASSWORD キーワードを指定することにより、開始タスクのユーザー ID が必ず保護されるようにしてください。

- RSE サーバーが固有の OMVS uid を持つようにしてください (その uid へ付与される z/OS UNIX 関連の特権のため)。
- RSE デーモンを適切に動作させるためには、大きなサイズのアドレス・スペース (2 GB) が必要です。この値は、ユーザー ID STCRSE の OMVS セグメントの ASSIZEMAX 変数で設定してください。この値を設定することにより、SYS1.PARMLIB(BPXPRMxx) で MAXASSIZE が変更されても、RSE デーモンには必要な領域サイズが確実に設定されます。
- RSE を適切に動作させるためには、多数のスレッドも必要です。スレッド数の限度は、ユーザー ID STCRSE の OMVS セグメントの THREADSMAX 変数で設定できます。限度の設定により、SYS1.PARMLIB(BPXPRMxx) で MAXTHREADS または MAXTHREADTASKS が変更されても、RSE には必要なスレッド数の限度が確実に設定されます。スレッド数の限度の適切な値を特定するには、「*Host Configuration Reference*」(SA88-4226) の『チューニングに関する考慮事項』を参照してください。
- ユーザー ID STCJMON も、OMVS セグメントで THREADSMAX を設定する対象となります。これは、JES ジョブ・モニターがクライアント接続ごとにスレッドを使用するためです。
- 統合デバッガー開始タスク (DBGMR) は、オプションの統合デバッガー・フィーチャーによってのみ使用されます。

STCRSE ユーザー ID の制限を考慮してください。RESTRICTED 属性を持つユーザーは、保護された (MVS) リソースには、特にアクセスを許可されている場合以外、アクセスできません。

ALTUSER STCRSE RESTRICTED

制限されたユーザーが「その他の」許可ビットによって z/OS UNIX ファイル・システム・リソースにアクセスできないよう、UNIXPRIV クラス内に RESTRICTED.FILESYS.ACCESS プロファイルを UACC(NONE) で定義します。ユーザー ID を制限する方法の詳細については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

重要: 制限されたユーザー ID を使用している場合は、TSO の **PERMIT** コマンドか、z/OS UNIX **setfac1** コマンドを使用して、リソースにアクセスする許可を明示的に追加します。リソースには、Developer for System z 資料が UACC を使用するリソース (PROGRAM クラス内の ** プロファイルなど)、またはそれが共通の z/OS UNIX 規則 (全員が Java ライブラリーの読み取り権限と実行権限を持つなど) に依存するリソースが含まれます。実動システム上でアクティブにする前にアクセスのテストをしてください。

セキュアな z/OS UNIX サーバーとして RSE を定義する

RSE は、クライアントのスレッドのセキュリティー環境を作成または削除するためには、BPX.SERVER プロファイルに対する UPDATE アクセス権を必要とします。このプロファイルが定義されていない場合は、UID(0) が RSE に必要です。このステップは、クライアントが接続可能になるために必要です。

統合デバッガーは、デバッグ・スレッドのセキュリティー環境を作成または削除するため、BPX.SERVER プロファイルに対する UPDATE アクセス権を必要とします。

このプロファイルが定義されない場合、STCDBM 開始タスク・ユーザー ID 用に UID(0) が必要です。この許可は、オプションの統合デバッガー・フィーチャーが使用される場合にのみ必要です。

- RDEFINE FACILITY BPX.SERVER UACC(NONE)
- PERMIT BPX.SERVER CLASS(FACILITY) ACCESS(UPDATE) ID(STCRSE)
- PERMIT BPX.SERVER CLASS(FACILITY) ACCESS(UPDATE) ID(STCDBM)
- SETROPTS RACLIST(FACILITY) REFRESH

重要: BPX.SERVER プロファイルを定義すると、z/OS UNIX 全体が UNIX レベルのセキュリティから、より安全な z/OS UNIX レベルのセキュリティに切り替わります。この切り替えによって、他の z/OS UNIX アプリケーションと操作が影響を受ける場合もあります。セキュリティは、実動システム上でアクティブにする前にテストしてください。さまざまなセキュリティ・レベルの詳細については、「UNIX System Services 計画」(GA88-8639) を参照してください。

RSE の MVS プログラム制御ライブラリーを定義する

BPX.SERVER に対する権限を持つサーバーは、クリーンなプログラム制御環境で実行する必要があります。この要件は、RSE によって呼び出されるすべてのプログラムも、プログラムで制御する必要があることを意味します。MVS ロード・ライブラリーの場合、プログラム制御はセキュリティ・ソフトウェアによって管理されます。このステップは、クライアントが接続可能になるために必要です。

RSE は、システム (SYS1.LINKLIB)、言語環境プログラムのランタイム (CEE.SCEERUN*)、および ISPF の TSO/ISPF クライアント・ゲートウェイ (ISP.SISPLOAD) ロード・ライブラリーを使用します。

- RALTER PROGRAM ** UACC(READ) ADDMEM('SYS1.LINKLIB'//NOPADCHK)
- RALTER PROGRAM ** UACC(READ) ADDMEM('CEE.SCEERUN'//NOPADCHK)
- RALTER PROGRAM ** UACC(READ) ADDMEM('CEE.SCEERUN2'//NOPADCHK)
- RALTER PROGRAM ** UACC(READ) ADDMEM('ISP.SISPLOAD'//NOPADCHK)
- SETROPTS WHEN(PROGRAM) REFRESH

注:すでに PROGRAM クラス内に * プロファイルがある場合は、** プロファイルを使用しないでください。プロファイルは、セキュリティ・ソフトウェアによって使用される検索パスが、分かりにくく、複雑なものになります。その場合は、既存の * 定義と新しい ** 定義をマージする必要があります。** プロファイルを使用してください。これについては、「Security Server RACF セキュリティ管理者のガイド」(SA88-8613) に説明があります。

オプションのサービスを使用できるようにするには、以下の前提条件の追加ライブラリーがプログラムで制御されるようにする必要があります。このリストには、Developer for System z が対話する製品 (IBM File Manager など) に固有のデータ・セットは含まれていません。

- 代替 REXX ランタイム・ライブラリー (SCLM Developer Toolkit 用)
 - REXX.*.SEAGALT
- システム・ロード・ライブラリー (SSL 暗号化用)

– SYS1.SIEALNKE

- Developer for System z ライブラリー (統合デバッガー用)

– FEK.SFEKAUTH

注: LPA 配置用に設計されたライブラリーは、LINKLIST または STEPLIB によってアクセスされる場合、追加のプログラム制御権限も必要とします。この資料では、以下の LPA ライブラリーの使用方法について説明します。

- ISPF (TSO/ISPF クライアント・ゲートウェイ用)

– ISP.SISPLPA

- REXX ランタイム・ライブラリー (SCLM Developer Toolkit 用)

– REXX.*.SEAGLPA

- Developer for System z (CARMA 用)

– FEK.SFEKLPA

RSE の PassTicket サポートを定義する

クライアントのパスワードまたは、X.509 証明書などのその他の識別手段は、接続時に ID を検査するためにのみ使用されます。その後は、スレッド・セキュリティを維持するために PassTicket が使用されます。このステップは、クライアントが接続可能になるために必要です。

PassTicket は、有効期間が約 10 分のシステム生成パスワードです。パスチケットは、秘密鍵に基づいて生成されます。この鍵は 64 ビット番号 (16 個の 16 進文字) です。以下のサンプル RACF コマンドでは、key16 プレースホルダーを、0 から 9 までと A から F までの文字を持つユーザー指定の 16 文字の 16 進数ストリングに置き換えてください。

- ```
RDEFINE PTKTDATA FEKAPPL UACC(NONE) SSIGNON(KEYMASKED(key16))
APPLDATA('NO REPLAY PROTECTION – DO NOT CHANGE')
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
```
- ```
RDEFINE PTKTDATA IRRPTAUTH.FEKAPPL.* UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
```
- ```
PERMIT IRRPTAUTH.FEKAPPL.* CLASS(PTKTDATA) ACCESS(UPDATE) ID(STCRSE)
```
- ```
SETROPTS RACLIST(PTKTDATA) REFRESH
```

RSE は、FEKAPPL 以外のアプリケーション ID の使用をサポートしています。これをアクティブにするには、「*IBM Rational Developer for System z* ホスト構成ガイド」の『_RSE_JAVAOPTS での追加 Java 始動パラメーターの定義』で説明されているように、rsed.envvars 内の「APPLID=FEKAPPL」オプションをコメント解除してカスタマイズします。PTKTDATA クラス定義は、RSE が使用する実際のアプリケーション ID と一致している必要があります。

OMVSAPPL はアプリケーション ID として使用しないでください。これは、大部分の z/OS UNIX アプリケーションの秘密鍵を公開するからです。また、デフォルトの MVS アプリケーション ID (MVS の直後にシステムの SMF ID を続けたもの) も使

用しないでください。これは、大部分の MVS アプリケーション (ユーザー・バッチ・ジョブを含む) の秘密鍵を公開するからです。

注:

- PTKTDATA クラスがすでに定義されている場合は、上記のリストにあるプロファイルを作成する前に、それが総称クラスとして定義されていることを確認してください。PTKTDATA クラス内の総称文字のサポートは、PassTicket に Java インターフェースが導入された z/OS リリース 1.7 からの新機能です。
- RSE が PassTicket を生成できるユーザー ID を制限するには、IRRPTAUTH.FEKAPPL.* 定義の中のワイルドカード (*) を、有効なユーザー ID マスクで置き換えます。
- RACF の設定によっては、プロファイルを定義しているユーザーが、そのプロファイルのアクセス・リストにも入っている場合があります。PTKTDATA プロファイルのこの許可を削除してください。
- RSE が提示した PassTicket を JES ジョブ・モニターが評価できるようにするには、JES ジョブ・モニターと RSE が、同じアプリケーション ID を持っている必要があります。JES ジョブ・モニターの場合、アプリケーション ID は FEJJCNFG 構成ファイルで APPLID ディレクティブによって設定されます。
- システムに暗号製品がインストールされており、使用可能になっている場合、保護されたサインオン・アプリケーション鍵を暗号化して、保護を強化することができます。それを行うには、KEYMASKED ではなく KEYENCRYPTED キーワードを使用します。詳しくは、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

重要: パスチケットが正しくセットアップされていないと、クライアント接続要求は失敗します。

RSE のアプリケーション保護の定義

クライアントがログオンするときに、RSE デーモンはユーザーがアプリケーションの使用を許可されていることを検証します。

- ```
RDEFINE APPL FEKAPPL UACC(READ) DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
```
- ```
SETROPTS RACLIST(APPL) REFRESH
```

注:

- 54 ページの『RSE の PassTicket サポートを定義する』で詳しく説明するように、RSE は FEKAPPL 以外のアプリケーション ID の使用をサポートしています。APPL クラス定義は、RSE が使用する実際のアプリケーション ID と一致している必要があります。
- アプリケーション ID が APPL クラスに定義されていない場合、クライアント接続要求は成功します。
- アプリケーション ID が定義されていて、ユーザーがプロファイルに対する READ 権限を欠いている場合にのみ、クライアント接続要求は失敗します。

JES コマンド・セキュリティを定義する

JES ジョブ・モニターは、ユーザーが要求したすべての JES オペレーター・コマンドを、拡張 MCS (EMCS) コンソールを通じて発行します。このコンソールの名前は、「*IBM Rational Developer for System z* ホスト構成ガイド」の『FEJJCNFG - JES ジョブ・モニター構成ファイル』の説明にあるように、`CONSOLE_NAME` ディレクティブによって制御されます。

以下のサンプル RACF コマンドは、Developer for System z ユーザーに、JES コマンドの限定セット (保留、保留解除、キャンセル、およびパージ) に対する条件付きアクセス権を与えます。ユーザーは、JES ジョブ・モニターによってコマンドを発行した場合にのみ、実行権限を持ちます。`#console` プレースホルダーは、実際のコンソール名に置き換えてください。

```
•  
RDEFINE OPERCMDS MVS.MCSOPER.#console UACC(READ)  
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')  
•  
RDEFINE OPERCMDS JES%.** UACC(NONE)  
•  
PERMIT JES%.** CLASS(OPERCMDS) ACCESS(UPDATE) WHEN(CONSOLE(JMON)) ID(*)  
•  
SETROPTS RACLIST(OPERCMDS) REFRESH
```

注:

- コンソールの使用は、`MVS.MCSOPER.#console` プロファイルが定義されていない場合に許可されます。
- `WHEN(CONSOLE(JMON))` が機能するためには、`CONSOLE` クラスがアクティブでなければなりませんが、`CONSOLE` クラス内に `EMCS` コンソールがあるかどうかについての実際のプロファイル検査はありません。
- `WHEN(CONSOLE(JMON))` 文節内で、`JMON` を実際のコンソール名に置き換えしないでください。`JMON` キーワードは、コンソール名ではなく、入り口点アプリケーションを表しています。

重要: ご使用のセキュリティ・ソフトウェアで汎用アクセス `NONE` を使用して JES コマンドを定義すると、他のアプリケーションや操作に影響が出る場合があります。セキュリティは、実動システム上でアクティブにする前にテストしてください。

表 11 および 57 ページの表 12 は、JES2 および JES3 について発行されたオペレーター・コマンドと、それらを保護するために使用できる個別セキュリティ・プロファイルを示しています。

表 11. JES2 ジョブ・モニターのオペレーター・コマンド

アクション	コマンド	OPERCMDS プロファイル	必要なアクセス権
保留	\$Hx(jobid) x = {J, S, または T}	jesname.MODIFYHOLD.BAT jesname.MODIFYHOLD.STC jesname.MODIFYHOLD.TSU	UPDATE

表 11. JES2 ジョブ・モニターのオペレーター・コマンド (続き)

アクション	コマンド	OPERCMDS プロファイル	必要なアクセス権
保留解除	\$Ax(jobid) x = {J、S、または T}	jesname.MODIFYRELEASE.BAT jesname.MODIFYRELEASE.STC jesname.MODIFYRELEASE.TSU	UPDATE
キャンセル	\$Cx(jobid) x = {J、S、または T}	jesname.CANCEL.BAT jesname.CANCEL.STC jesname.CANCEL.TSU	UPDATE
パージ	\$Cx(jobid),P x = {J、S、または T}	jesname.CANCEL.BAT jesname.CANCEL.STC jesname.CANCEL.TSU	UPDATE

表 12. JES3 ジョブ・モニターのオペレーター・コマンド

アクション	コマンド	OPERCMDS プロファイル	必要なアクセス権
保留	*F,J=jobid,H	jesname.MODIFY.JOB	UPDATE
保留解除	*F,J=jobid,R	jesname.MODIFY.JOB	UPDATE
キャンセル	*F,J=jobid,C	jesname.MODIFY.JOB	UPDATE
パージ	*F,J=jobid,C	jesname.MODIFY.JOB	UPDATE

注:

- 「保留」、「保留解除」、「キャンセル」、「パージ」の各 JES オペレーター・コマンドと「JCL の表示」コマンドは、クライアント・ユーザー ID が所有しているスプール・ファイルに対してのみ実行できます。ただし、JES ジョブ・モニター構成ファイル内で、LIMIT_COMMANDS= が値 LIMITED または NOLIMIT に指定されている場合は除きます。詳しくは、「ホスト構成リファレンス」(SA88-4226) の『ジョブに対するアクション - ターゲットの制限』を参照してください。
- ユーザーは、JES ジョブ・モニター構成ファイル内で LIMIT_VIEW=USERID が定義されている場合を除き、すべてのスプール・ファイルを参照できます。詳しくは、「ホスト構成リファレンス」(SA88-4226) の『スプール・ファイルへのアクセス』を参照してください。
- ユーザーにこれらのオペレーター・コマンドの許可がない場合でも、これらのリソースを保護できるプロファイル (JESINPUT、JESJOBS、および JESSPOOL クラス内のプロファイルなど) に対して十分な権限を持っていれば、JES ジョブ・モニターを通じてジョブを実行依頼し、ジョブ出力を読み取ることができます。

TSO セッションから JMON コンソールを作成することによって JES ジョブ・モニター・サーバーの ID を装うことは、セキュリティ・ソフトウェアによって防止されます。コンソールを作成できても、例えば、JES ジョブ・モニターと TSO とでは、エントリー・ポイントが異なります。この資料で説明されているとおりにセキュリティがセットアップされており、ユーザーが他の手段によって JES コマンドに対する権限を持っていない場合は、そのコンソールから発行された JES コマンドはセキュリティ検査で不合格になります。

データ・セット・プロファイルを定義する

ほとんどの Developer for System z データ・セットの場合、ユーザーには READ アクセス権限、システム・プログラマーには ALTER アクセス権限で十分です。

#sysprog プレースホルダーは、有効なユーザー ID または RACF グループ名に置き換えてください。また、正しいデータ・セット名については、製品をインストールおよび構成したシステム・プログラマーに問い合わせてください。FEK は、インストール時に使用されたデフォルトの高位修飾子で、FEK.#CUST はカスタマイズ・プロセスで作成されたデータ・セットのデフォルトの高位修飾子です。

```
•  
  ADDGROUP (FEK) OWNER(IBMUSER) SUPGROUP(SYS1)  
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - HLQ STUB')  
•  
  ADDSD 'FEK.*.**' UACC(READ)  
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')  
•  
  PERMIT 'FEK.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)  
•  
  SETROPTS GENERIC(DATASET) REFRESH
```

注:

- このデータ・セットは APF 許可されているため、更新に対して FEK.SFEKAUTH を保護します。これは、FEK.SFEKLOAD と FEK.SFEKLPA についても当てはまります。これらのデータ・セットがプログラムで制御されるためです。
- この資料内および FEKRACF ジョブ内のサンプル・コマンドは、Enhanced Generic Naming (EGN) がアクティブであることを想定しています。EGN がアクティブであるとき、** 修飾子を使用して、DATASET クラス内の任意の数の修飾子を表すことができます。使用しているシステムで EGN がアクティブでない場合は、** を * に置き換えてください。EGN の詳細については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

一部のオプションの Developer for System z コンポーネントには、追加のセキュリティー・データ・セット・プロファイルが必要です。#sysprog、#ram-developer、および #cicsadmin の各プレースホルダーは、有効なユーザー ID または RACF グループ名に置き換えてください。

- SCLM Developer Toolkit のロング/ショート・ネーム変換を使用している場合は、ユーザーにマッピング VSAM の FEK.#CUST.LSTRANS.FILE に対する UPDATE アクセス権が必要です。

```
—  
  ADDSD 'FEK.#CUST.LSTRANS.*.**' UACC(UPDATE)  
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - SCLMDT')  
—  
  PERMIT 'FEK.#CUST.LSTRANS.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)  
—  
  SETROPTS GENERIC(DATASET) REFRESH
```

- CARMA RAM (Repository Access Manager) 開発者には、CARMA VSAM である FEK.#CUST.CRA* に対する UPDATE アクセス権が必要です。

```

-
ADDSD 'FEK.#CUST.CRA*.*' UACC(READ)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - CARMA')
-
PERMIT 'FEK.#CUST.CRA*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
-
PERMIT 'FEK.#CUST.CRA*.*' CLASS(DATASET) ACCESS(UPDATE) ID(#ram-developer)
-
SETROPTS GENERIC(DATASET) REFRESH

```

- Application Deployment Manager の CICS リソース定義 (CRD) サーバーを使用している場合は、CICS 管理者に、CRD リポジトリ VSAM に対する UPDATE アクセス権が必要です。

```

-
ADDSD 'FEK.#CUST.ADNREP*.*' UACC(READ)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
-
PERMIT 'FEK.#CUST.ADNREP*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
-
PERMIT 'FEK.#CUST.ADNREP*.*' CLASS(DATASET) ACCESS(UPDATE) ID(#cicsadmin)
-
SETROPTS GENERIC(DATASET) REFRESH

```

- Application Deployment Manager のマニフェスト・リポジトリが定義されている場合、すべての CICS Transaction Server ユーザーに、マニフェスト・リポジトリ VSAM に対する UPDATE アクセス権が必要です。

```

-
ADDSD 'FEK.#CUST.ADNMAN*.*' UACC(UPDATE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
-
PERMIT 'FEK.#CUST.ADNMAN*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
-
SETROPTS GENERIC(DATASET) REFRESH

```

READ アクセス権も制御対象とする、さらにセキュアなセットアップのためには、以下のサンプル RACF コマンドを使用します。

- uacc(none) データ・セット保護

```

-
ADDGROUP (FEK)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - HLQ STUB')
OWNER(IBMUSER) SUPGROUP(SYS1)"
-
ADDSD 'FEK.*.*' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
-
ADDSD 'FEK.SFEKAUTH' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')

```

```

-
ADDSD 'FEK.SFEKLOAD' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')

-
ADDSD 'FEK.SFEKLMOD' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')

-
ADDSD 'FEK.SFEKPROC' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')

-
ADDSD 'FEK.#CUST.PARMLIB' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')

-
ADDSD 'FEK.#CUST.CNTL' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')

-
ADDSD 'FEK.#CUST.SQL' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')

-
ADDSD 'FEK.#CUST.LSTRANS.*.**' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - SCLMDT')

-
ADDSD 'FEK.#CUST.CRA*.**' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - CARMA')

-
ADDSD 'FEK.#CUST.ADNREP*.**' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')

-
ADDSD 'FEK.#CUST.ADNMAN*.**' UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')

```

- システム・プログラマーがすべてのライブラリーを管理することを許可する

```

-
PERMIT 'FEK.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

-
PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

-
PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

-
PERMIT 'FEK.SFEKLMOD' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

-
PERMIT 'FEK.SFEKPROC' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

-
PERMIT 'FEK.#CUST.PARMLIB' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

-
PERMIT 'FEK.#CUST.CNTL' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

-
PERMIT 'FEK.#CUST.SQL' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

```

```

-
PERMIT 'FEK.#CUST.LSTRANS.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

```

```

-
PERMIT 'FEK.#CUST.CRA*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

```

```

-
PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

```

```

-
PERMIT 'FEK.#CUST.ADNMAN*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

```

- クライアントがロード・ライブラリーおよび exec ライブラリーにアクセスすることを許可する

```

-
PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(READ) ID(*)

```

```

-
PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(READ) ID(*)

```

```

-
PERMIT 'FEK.SFEKPROC' CLASS(DATASET) ACCESS(READ) ID(*)

```

```

-
PERMIT 'FEK.#CUST.CNTL' CLASS(DATASET) ACCESS(READ) ID(*)

```

```

-
PERMIT 'FEK.#CUST.SQL' CLASS(DATASET) ACCESS(READ) ID(*)

```

注: LPA 内に存在するすべてのコードには、全員がアクセスできるため、FEK.SFEKLPA に許可は必要ありません。

- 統合デバッガーにロード・ライブラリーへのアクセスを許可する

```

- PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(READ) ID(STCDBM)

```

- JES ジョブ・モニターにロードおよびパラメーター・ライブラリーへのアクセスを許可する

```

-
PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(READ) ID(STCJMON)

```

```

-
PERMIT 'FEK.#CUST.PARMLIB' CLASS(DATASET) ACCESS(READ) ID(STCJMON)

```

- (オプション) クライアントに SCLMDT 用のロング/ショート・ネーム変換 VSAM 更新を許可する

```

-
PERMIT 'FEK.#CUST.LSTRANS.*.**' CLASS(DATASET) ACCESS(UPDATE) ID(*)

```

- (オプション) RAM 開発者に CARMA 用の CARMA VSAM の更新を許可する

```

-
PERMIT 'FEK.#CUST.CRA*.**' CLASS(DATASET) ACCESS(UPDATE) ID(#ram-developer)

```

- (オプション) CICS ユーザーに、Application Deployment Manager 用の CRD リポジトリ VSAM の読み取りを許可する

```

-
PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(READ) ID(*)

```


- (オプション) CICS 管理者に、Application Deployment Manager 用の CRD リポジトリ VSAM の更新を許可する

—

```
PERMIT 'FEK.#CUST.ADNREP*.*' CLASS(DATASET) ACCESS(UPDATE) ID(#cicsadmin)
```

- (オプション) CICS ユーザーに、Application Deployment Manager 用のマニフェスト・リポジトリ VSAM の更新を許可します。

—

```
PERMIT 'FEK.#CUST.ADNMAN*.*' CLASS(DATASET) ACCESS(UPDATE) ID(*)
```

- (オプション) CICS TS サーバーに bidi 用のロード・ライブラリーと Application Deployment Manager へのアクセスを許可する

—

```
PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(READ) ID(#cicsts)
```

- (オプション) CICS TS サーバー、IMS™ 領域、および MVS バッチ・ジョブに、IRZ メッセージ用のロード・ライブラリーへのアクセスを許可する

—

```
PERMIT 'FEK.SFEKLMOD' CLASS(DATASET) ACCESS(READ) ID(#cicsts)
PERMIT 'FEK.SFEKLMOD' CLASS(DATASET) ACCESS(READ) ID(#ims)
PERMIT 'FEK.SFEKLMOD' CLASS(DATASET) ACCESS(READ) ID(#batch)
```

- セキュリティー・プロファイルをアクティブにする

—

```
SETROPTS GENERIC(DATASET) REFRESH
```

システム・データ・セットへの READ アクセスを制御するときは、Developer for System z サーバーおよびユーザーに以下のデータ・セットに対する READ 権限を与える必要があります。

- CEE.SCEERUN
- CEE.SCEERUN2
- CBC.SCLBDLL
- ISP.SISPLD
- ISP.SISPLPA
- SYS1.LINKLIB
- SYS1.SIEALNKE
- SYS1.SIEAMIGE
- REXX.V1R4M0.SEAGLPA

注: REXX 製品パッケージに代替ライブラリーを使用する場合、デフォルト REXX ランタイム・ライブラリー名は、前述の例で使用されている REXX.*.SEAGLPA ではなく、REXX.*.SEAGALT となります。

RSE の z/OS UNIX プログラム制御ファイルを定義する

BPX.SERVER に対する権限を持つサーバーは、クリーンなプログラム制御環境で実行する必要があります。この要件は、RSE によって呼び出されるすべてのプログラムも、プログラムで制御する必要があることを意味します。z/OS UNIX ファイルの場合、プログラム制御は **extattr** コマンドによって管理されます。このコマンドを実

行するには、FACILITY クラス内の BPX.FILEATTR.PROGCTL に対する READ アクセス権を持つか、または UID(0) であることが必要です。

RSE サーバーは、RACF の Java 共用ライブラリー (/usr/lib/libIRRRacf*.so) を使用します。

- extattr +p /usr/lib/libIRRRacf*.so

注:

- z/OS 1.9 以降、/usr/lib/libIRRRacf*.so は SMP/E RACF のインストール中に、プログラムによる制御モードでインストールされます。
- z/OS 1.10 以降、/usr/lib/libIRRRacf*.so はベース z/OS に付属の SAF の一部であるので、RACF 以外のお客様にもご利用いただけます。
- RACF 以外のセキュリティ製品を使用している場合は、セットアップが異なることがあります。詳しくは、ご使用のセキュリティ製品の資料を参照してください。
- Developer for System z の SMP/E インストールは、内部 RSE プログラムのプログラム制御ビットを設定します。
- プログラム制御ビットの現在の状況を表示するには、z/OS UNIX コマンド **ls -Eog** を使用します。2 番目のストリング内に英字の **p** が表示される場合、そのファイルはプログラムで制御されます。

```
$ ls -Eog /usr/lib/libIRRRacf*.so
-rwxr-xr-x  aps- 2      69632 Oct  5 2007 /usr/lib/libIRRRacf.so
-rwxr-xr-x  aps- 2      69632 Oct  5 2007 /usr/lib/libIRRRacf64.so
```

セキュリティ設定の検査

セキュリティに関連するカスタマイズの結果を表示するには、以下のサンプル・コマンドを使用します。

- セキュリティの設定とクラス
 - SETROPTS LIST
- ユーザーの OMVS セグメント
 - LISTUSER #userid NORACF OMVS
 - LISTGRP #group-name NORACF OMVS
- 開始タスク
 - LISTGRP STCGROUP OMVS
 - LISTUSER STCDBM OMVS
 - LISTUSER STCJMON OMVS
 - LISTUSER STCRSE OMVS
 - RLIST STARTED DBGMR.* ALL STDATA
 - RLIST STARTED JMON.* ALL STDATA
 - RLIST STARTED RSED.* ALL STDATA
- セキュアな z/OS UNIX サーバーとしての RSE
 - RLIST FACILITY BPX.SERVER ALL
- RSE の MVS プログラム制御ライブラリー
 - RLIST PROGRAM ** ALL

- RSE の PassTicket サポート
 - RLIST PTKTDATA FEKAPPL ALL SSIGNON
 - RLIST PTKTDATA IRRPTAUTH.FEKAPPL.* ALL
- RSE のアプリケーション保護
 - RLIST APPL FEKAPPL ALL
- JES コマンド・セキュリティー
 - RLIST CONSOLE JMON ALL
 - RLIST OPERCMDS MVS.MCSOPER.JMON ALL
 - RLIST OPERCMDS JES%.** ALL
- データ・セット・プロファイル
 - LISTGRP FEK
 - LISTDSD PREFIX(FEK) ALL
- RSE の z/OS UNIX プログラム制御ファイル
 - ls -E /usr/lib/libIRRRacf*.so

オプションで、特定のユーザーに対して Developer for System z の動作を指定するプロファイルが存在できます。それらのプロファイルは、FEK.** フィルターに一致し、デフォルトで FACILITY クラスに置かれます。rsed.envvars の `_RSE_FEK_SAF_CLASS` ディレクティブを参照してください。**SEARCH** コマンドを使用してプロファイル名をリストすることができます。プロファイルの詳細を表示するには、**RLIST** コマンドを使用します。

- SEARCH CLASS(FACILITY) FILTER(FEK.**)
- RLIST FACILITY #プロファイル名 ALL

第 3 章 TCP/IP に関する考慮事項

Developer for System z では、TCP/IP を使用して、非メインフレーム・ワークステーションのユーザーに、メインフレームからアクセスすることができます。また、各種コンポーネントと他の製品との通信にも TCP/IP が使用されます。

ほとんどの Developer for System z 機能は z/OS UNIX をベースにしているため、TCP/IP は z/OS UNIX 検索順序を使用して、構成ファイルを検索します。詳しくは、221 ページの『第 14 章 TCP/IP のセットアップ』を参照してください。

この章では、以下のトピックについて説明します。

- 『TCP/IP ポート』
- 68 ページの『TCP/IP のデフォルト動作のオーバーライド』
- 68 ページの『複数スタック (CINET)』
- 70 ページの『Distributed Dynamic VIPA』

TCP/IP ポート

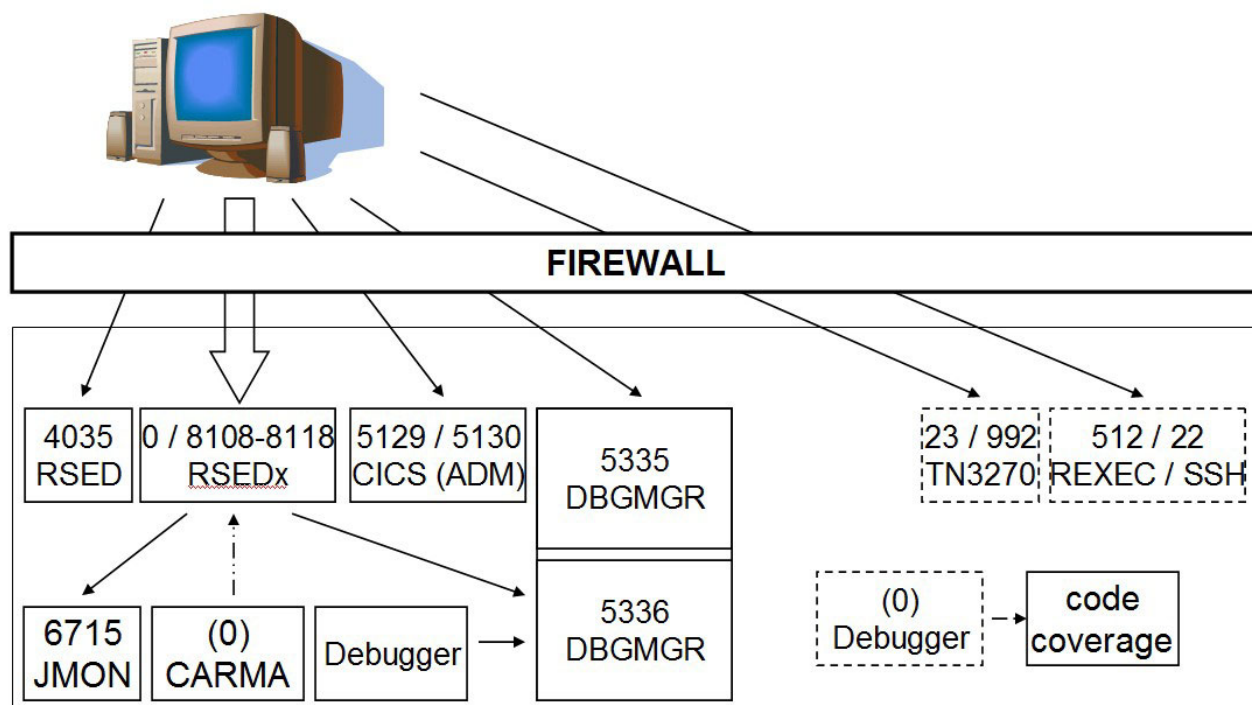


図 10. TCP/IP ポート

図 10 は、Developer for System z で使用できる TCP/IP ポートを示しています。矢印は、バインドの実行元 (矢印の先) と接続元を示しています。

外部通信

z/OS ホストを保護しているファイアウォールに対して、以下のポートを定義してください。これらのポートは、クライアント/ホスト通信 (tcp プロトコルを使用) に使用されるためです。

- クライアント/ホスト通信セットアップ用の RSE デーモン、デフォルト・ポート 4035。このポートは、rsed.envvars 構成ファイルの中で設定できます。このポート上の通信は、SSL または TLS を使用して暗号化できます。
- クライアント/ホスト通信用の RSE サーバー。デフォルトでは、使用可能な任意のポートを使用できますが、これは rsed.envvars 内の _RSE_PORTRANGE 定義によって、指定する範囲に制限できます。_RSE_PORTRANGE 用のデフォルトのポート範囲は 8108-8118 (11 ポート) です。このポート上の通信は、SSL または TLS を使用して暗号化できます。
- (オプション) 統合デバッガー・サービスのデバッグ・マネージャー、デフォルト・ポート 5335。このポートは DBGMR 開始タスク JCL で設定できます。このポート上の通信は、SSL または TLS を使用して暗号化できます。
- (オプション) z/OS UNIX サブプロジェクトのリモート (ホスト・ベースの) アクション用の以下のいずれかの INETD サービス。
 - REXEC (z/OS UNIX バージョン)、デフォルト・ポート 512。
 - SSH (z/OS UNIX バージョン)、デフォルト・ポート 22。このポート上の通信は、SSL を使用して暗号化されます。
- (オプション) Host Connect Emulator 用の TN3270 Telnet サービス、デフォルト・ポート 23。通信は、SSL または TLS を使用して暗号化できます (デフォルト・ポート 992)。TN3270 Telnet サービスに割り当てられるデフォルト・ポートは、ユーザーが暗号化の使用を選択するかどうかによって決まります。
- (オプション) Application Deployment Manager 用の、以下の CICSTS アプリケーション・インターフェース (いずれかまたは両方)
 - RESTful インターフェース、デフォルト・ポート 5130。このポートは CICS CSD で設定できます。
 - Web サービス・インターフェース、デフォルト・ポート 5129。このポートは CICS CSD で設定できます。このポート上の通信は、SSL を使用して暗号化できます。

注: 通常、ホストへの接続に使用する TCP/IP アドレスはクライアントが指定します。ただし、デバッグ・セッションが正しいホストと通信することを保証するため、デバッグ・マネージャーは使用しなければならない TCP/IP アドレスをクライアントに指示します。

内部通信

いくつかの Developer for System z ホスト・サービスは別のスレッドまたはアドレス・スペースで実行され、TCP/IP ソケットを通信メカニズムとして使用します。これらすべてのサービスは、クライアントとの通信に RSE を使用し、データ・ストリームをホストだけに限定します。一部のサービスでは、使用可能な任意のポートが使用され、それ以外のサービスでは、使用されるポートまたはポート範囲をシステム・プログラマーが選択できます。

- JES 関連サービスの JES ジョブ・モニター、デフォルト・ポート 6715。このポートは、FEJJCNFG 構成メンバーの中で設定可能であり、rsed.envvars 構成ファイルの中で繰り返されます。
- (オプション) デフォルトでは、CARMA 通信は一時ポートを使用しますが、ポート範囲は CRASRV.properties 構成ファイルの中で設定できます。
- (オプション) デバッグ関連サービスのデバッグ・マネージャー、デフォルト・ポート 5336。このポートは DBGMGR 開始タスク JCL で設定できます。
- ホスト・ベースのコード・カバレッジは、一時ポートを割り振って、統合デバッガーと通信して、コード・カバレッジ・レポートに必要なデータを送信できるようにするバッチ・ジョブです。

TCP/IP ポートの予約

Developer for System z が使用するポートを予約するのに PROFILE.TCPIP 内の PORT ステートメントまたは PORTRANGE ステートメントを使用する場合、RSE スレッド・プール内でアクティブなスレッドによって多数のバインドが行われることに注意してください。RSE スレッド・プールのジョブ名は、RSEDx です。ここで、RSED は RSE 開始タスクの名前で、x はランダムな 1 桁の数値です。したがって、定義内にワイルドカードが必要です。

```
PORT      4035      TCP RSED      ; Developer for System z - RSE daemon
PORT      6715      TCP JMON      ; Developer for System z - JES job monitor
PORT      5335      TCP DBGMGR    ; Developer for System z - Integrated
debugger
PORT      5336      TCP DBGMGR    ; Developer for System z - Integrated
debugger
PORTRange 8108 11   TCP RSED*     ; Developer for System z - _RSE_PORTRANGE
;PORTRange 5227 100 TCP RSED*     ; Developer for System z - CARMA
```

CARMA と TCP/IP ポート

CARMA (Common Access Repository Manager) は、ホスト・ベースの Software Configuration Manager (SCM) (CA Endeavor® SCM など) にアクセスするために使用されます。ほとんどの場合、サーバーは RSE デーモンの場合と同様に、ポートにバインドし、接続要求を listen します。しかし CARMA は別の方法を使用します。これは、クライアントが接続要求を開始した時点で CARMA サーバーがまだアクティブでないためです。

クライアントから接続要求が送信されると、RSE スレッド・プール内でユーザー・スレッドとしてアクティブとなっている CARMA マイナーは、一時ポートを要求するか、CRASRV.properties 構成ファイルに指定されている範囲から空いているポートを見つけ、そのポートにバインドします。次にこのマイナーは、CARMA サーバーを始動してポート番号を渡します。これによって、サーバーは接続先のポートを認識します。サーバーが接続されると、クライアントはサーバーに要求を送信して結果を受信できるようになります。

TCP/IP の観点では、RSE (CARMA マイナー経由) がポートにバインドするサーバーであり、CARMA サーバーがそのポートに接続するクライアントです。

CARMA が使用するポート範囲を予約するのに PROFILE.TCPIP 内の PORT ステートメントまたは PORTRANGE ステートメントを使用する場合、CARMA マイナーが RSE スレッド・プール内でアクティブになっていることに注意してください。RSE

スレッド・プールのジョブ名は、RSEDx です。ここで、RSED は RSE 開始タスクの名前で、x はランダムな 1 桁の数値です。したがって、定義内にワイルドカードが必要です。

```
PORTRange 5227 100 RSED* ; Developer for System z - CARMA
```

LDAP の考慮事項

RSE サーバーは、1 つ以上の LDAP サーバーに対して Developer for System z の各種サービスについて照会するように構成できます。

- LDAP グループに対して、クライアントへのプッシュの複数開発者グループ・サポートについて照会する。
- 1 つ以上の証明書失効リスト (CRL) に対して、X.509 認証について照会する。

TCP/IP のセキュリティー保護手段 (ファイアウォールなど) によって、(ホストベースの) RSE サーバーが LDAP サーバーと通信できなくなることがある点に注意してください。以下の情報を使用して、確実に LDAP サーバーにアクセスできるようにしてください。

- LDAP サーバーの TCP/IP アドレスまたは DNS 名は、rsed.envvars の *_LDAP_SERVER 変数にリストされています。
- LDAP サーバーのポート番号は、rsed.envvars の *_LDAP_PORT 変数にリストされています。
- LDAP は、TCP プロトコルを使用します。
- LDAP サーバーは、ホストベースの RSE サーバーによってアクセスされます。
- RSE サーバーは、RSEDx アドレス・スペース内でアクティブになります (RSED は RSE 開始タスクの名前になります。x はランダムな 1 桁の数値になります。例えば、RSED8 となります)。

TCP/IP のデフォルト動作のオーバーライド

遅延 ACK

遅延 ACK では、TCP パケットの受信確認応答を最大 200 ミリ秒遅らせます。この遅延により、受信したパケットに対する応答と共に ACK を送信できる機会が多くなり、ネットワーク・トラフィックを削減できるようになります。ただし、送信側が ACK を受信するまで新規パケットを送信しない場合 (例えば、Nagle アルゴリズムを実装したため) に、送信されたパケットに対する応答が存在しないと (例えば、そのパケットがファイル転送の一部である場合)、通信に不必要な遅延を招きます。

Developer for System z では、遅延 ACK 機能を使用不可にできます。これを実行するには、ホスト側の rsed.envvars にある DSTORE_TCP_NO_DELAY ディレクティブを使用します。詳しくは「ホスト構成ガイド」(SC88-5663) を参照してください。

複数スタック (CINET)

z/OS Communication Server を使用すると、単一のシステム上で同時に複数の TCP/IP スタックをアクティブにすることができます。これは、CINET セットアップと呼ばれます。

デフォルトのスタック上で Developer for System z がアクティブになっていない場合、選択した Developer for System z 機能は失敗することがあります。スタックのアフィニティーを使用することがこの問題を解決する確実な方法です。スタックのアフィニティーは、使用可能なすべての TCP/IP スタック（これは開始タスクのデフォルトです）ではなく、特定の TCP/IP スタックのみを使用するように Developer for System z に指示します。

スタックのアフィニティーは、rsed.envvars 構成ファイル内の `_BPXK_SETIBMOPT_TRANSPORT` ディレクティブをコメント解除およびカスタマイズすることにより、RSED 開始タスク用に設定されます。この構成ファイルのカスタマイズの詳細については、「『ホスト構成ガイド』」(SC88-5663) の『第 2 章 基本的なカスタマイズ』内の関連するセクションを参照してください。

CARMA とスタックのアフィニティー

CARMA (Common Access Repository Manager) は、ホスト・ベースの Software Configuration Manager (SCM) (CA Endeavor® SCM など) にアクセスするために使用されます。これを行うために、CARMA はユーザー固有のサーバーを始動します。したがって、スタックのアフィニティーを強制的に設定するために追加の構成が必要になります。

Developer for System z 開始タスクと同様に、CARMA サーバーのスタックのアフィニティーは、`_BPXK_SETIBMOPT_TRANSPORT` 変数を使用して設定されます。この変数は LE (言語環境プログラム) に渡される必要があります。これは、アクティブな `crastart*.conf` または `CRASUB*` 構成ファイル内の始動コマンドを調整することにより実行可能です。

注:

- 始動コマンドを保持する構成ファイルの正確な名前は、CARMA を構成したシステム・プログラマーによるさまざまな選択によって異なります。これについて詳しくは、「『ホスト構成ガイド』」(SC88-5663) の『第 3 章 (オプション) 共通アクセス・リポジトリ・マネージャー (CARMA)』を参照してください。
- `_BPXK_SETIBMOPT_TRANSPORT` は、使用される TCP/IP スタックの名前を指定します。この名前は、関連する `TCPIP.DATA` 内の `TCPIPJOBNAME` ステートメントで定義されています。
- `SYSTCPD DD` ステートメントのコーディングでは、要求されたスタックのアフィニティーは設定されません。
- デフォルトでは、CARMA は通常の TCP/IP スタックを使用しません。CARMA は、CARMA マイナーと CARMA サーバーの間の通信にループバック・アドレスを使用します。これによって、セキュリティが向上し (ローカル・プロセスだけがループバック・アドレスにアクセスできるようにする)、CARMA 通信にスタック・アフィニティーを追加する必要がなくなる可能性が高くなります。

crastart*.conf

以下の部分を置き換えます。

```
... PARM(&CRAPRM1. &CRAPRM2.)
```

これを以下と置き換えてください (ここで、TCPIP は、希望する TCP/IP スタックを表します)。

```
... PARM(ENVAR("_BPXK_SETIBMOPT_TRANSPORT=TCPIP") / &CRAPRM1. &CRAPRM2.)
```

注: CRASTART は、行継続をサポートしないが、受け入れられる行の長さに制限はありません。

CRASUB*

以下の部分を置き換えます。

```
... PARM(&PORT &TIMEOUT)
```

これを以下と置き換えてください (ここで、TCPIP は、希望する TCP/IP スタックを表します)。

```
... PARM(ENVAR("_BPXK_SETIBMOPT_TRANSPORT=TCPIP") / &PORT &TIMEOUT)
```

注: ジョブ実行依頼では、行の長さが 80 文字に制限されます。これより長い行はブランク () で改行し、1 行目の最後に正符号 (+) を使用して、2 つの行を連結します。

Distributed Dynamic VIPA

Distributed DVIPA (Dynamic Virtual IP Addressing) では、ご使用のシスプレックス内の別々のシステムで、同一の Developer for System z セットアップを同時に実行し、また、オプションで、WLM を使用して、TCP/IP に、これらのシステム間でクライアント接続を分散させることができます。

Distributed DVIPA を構成する方法はいくつかありますが、Developer for System z はこれらのオプションにいくつかの制限を設けています。

- RSE デーモンは Distributed DVIPA に定義されたポートを所有していますが、実際の動作は RSE サーバーで発生します。これは別のアドレス・スペースのスレッドとしてアクティブです。したがって、ご使用のシステムでロード・バランシングを行うために SERVERWLM 分散メソッドを使用することはできません。これは、WLM が、RSE サーバーではなく RSE デーモンの統計に基づいてアドバイスを行うためです。
- クライアントは、RSE デーモン用シスプレックス・ディストリビューターが使用する DVIPA アドレスしか認識していません。シスプレックス・ディストリビューターは、接続要求を、使用可能な RSE デーモンの 1 つに受け渡します。するとこのデーモンは、システム上のポートにバインドする RSE サーバー・スレッドを開始します。クライアントがこのポートに接続するときは、実際のシステム・アドレスではなく、もう一度 DVIPA アドレスを使用します。このため、シスプレックス・ディストリビューターが必ず新しい接続を正しいシステムにリダイレクトするようにする必要があります。

このため、Developer for System z は、RSE サーバー・スレッドが使用するポートが必ずシスプレックス内で固有であるようにするために、VIPADISTRIBUTE ステートメントに SYSPLEXPORTS を定義する必要があります。

注:

- SYSPLEXPORTS を使用することは、EZBEPOR 構造体のご使用のカップリング・ファシリティーで定義されていなければならないことを暗黙に示しています。

- SYSPLEXPORTS を使用することは、TCP/IP が 2 次接続に一時ポートを選択することを暗黙に示しています。これは、TCP/IP プロファイルのこれらの接続に対して、PORT ディレクティブと PORTRANGE ディレクティブを使用してポートを予約することができないことを暗黙に示しています。また、rsed.envvars 内の _RSE_PORTRANGE を使用して、Developer for System z が使っているポートを制限することはできません。これによりファイアウォールが複雑化してしまうため、Developer for System z は、このような制限に対し回避策を提示します。

Distributed DVIPA を使用するとき、Developer for System z の中でもいくつかの制限があります。

- Developer for System z クライアントが、TCP/IP による正しいポート選択に干渉しないようにするには、rsed.envvars の deny.nonzero.port ディレクティブを有効にする必要があります。
- 参加するすべての Developer for System z サーバーは、同一のセットアップでなければなりません。/usr/lpp/rdz と /etc/rdz を、すべての参加システムで共有する必要があります。また、/var/rdz/projects、/var/rdz/pushtoclient、および /var/rdz/sc1mdt のディレクトリーを使用する場合は、これらのディレクトリーも共有する必要があります。/var/rdz/WORKAREA と /var/rdz/logs は、各システムで固有でなければならないことに注意してください。
- どの Developer for System z コンポーネントが共有されなければならないのか、また、どのコンポーネントがシステムごとに固有でなければならないのかに関しては、177 ページの『第 11 章 実行、複数のインスタンスの』を参照してください。

JES ジョブ・モニター、CARMA、および他の Developer for System z サーバーは、ローカル RSE としか対話しないため、DVIPA のセットアップを必要としません。

統合デバッガーはローカル RSE と対話するので、DVIPA のセットアップを必要としません。デバッグ・セッションが正しいホストと通信することを保証するため、デバッグ・マネージャーは使用しなければならない TCP/IP アドレスをクライアントに指示するので、DVIPA のセットアップは必要としません。

Distributed DVIPA は、ご使用の TCP/IP プロファイルの VIPADynamic ブロックの VIPADefine キーワードと VIPABackup キーワードで定義されます。VIPADISTribute キーワードは、必要なシスプレックス・ディストリビューター定義を追加します。Distributed DVIPA では、参加しているすべてのスタックがシスプレックスを認識する必要があります。これは、ご使用の TCP/IP プロファイルの IPCONFIG ブロックの SYSPLExRouting キーワードと DYNAMICXCF キーワードを経由して行われます。これらのディレクティブの詳細については、「*Communications Server IP 構成解説書*」(SC88-8927)を参照してください。

ご使用のカップリング・ファシリティーに EZBEPORts 構造をセットアップする方法についての詳細は、「MVS シスプレックスのセットアップ」(SA88-8591) および「*Communication Server: SNA ネットワーク・インプリメンテーション・ガイド*」(SC88-8928)を参照してください。

ポート選択の制限

SYSPLXPORTS を使用することは、TCP/IP が 2 次接続に一時ポートを選択することを暗黙に示しています。一時ポートは、必ず、空き状態かつ予約がされていない状態でなくてはなりません。一時ポートを利用すると、ファイアウォールのベスト・プラクティスとの相性が悪く、通信用にオープンにオープンされるポートを制限してしまいます。その理由は、どのポートが使用されるか不明になるからです。

この問題を回避するには、Developer for System z に対しシステムごとに固有の `_RSE_PORTRANGE` を定義することで 2 次接続に既知のポートを強制的に割り当てます。その際使用されるポートの範囲は、すべてのシステム上で使用される Developer for System z 用に予約されていることを確認します。この回避方法を実行するには、TCP/IP APAR PM63379 が必須であることに注意してください。

TCP/IP が 2 次接続を正しいシステムに確実に経路指定するには、Developer for System z は各システム上で固有のポート範囲を使用する必要があります。このことは、そのシステムに共有された、同一のセットアップが使用できないということを暗に示しています。これは、`rsed.envvars` 内にある `_RSE_PORTRANGE` は、固有でなくてはならないからです。同じコードを利用する場合、異なる構成ファイルを使用して複数のサーバーをセットアップする方法については、177 ページの『第 11 章 実行、複数のインスタンスの』内、178 ページの『同一のソフトウェア・レベル、異なる構成ファイル』を参照してください。異なったシステムに対して同一のファイルを確実に使用するには、`rsed.envvars` のマスター・コピーと、システム固有のセットアップに合わせてその調整とコピーを行うためのスクリプトを使用する必要があります。

1. SYS1 上に、単一システム・セットアップのように Developer for System z をセットアップしてください。ただし、`/usr/lpp/rdz` および `/etc/rdz` は共有ファイル・システムに配置されるようにしてください。すべての MVS ベースのパーツは SYS2 と共有状態にある必要があります。
2. マスター・コピーとして `/etc/rdz/rsed.envvars` を使用し、ファイルの終わりに `/etc/rdz` への参照を追加してください。それによって、システム固有のコピーが残りの構成ファイルを選出できるようにします。

```
$ oedit /etc/rdz/rsed.envvars
-> add the following at the END:
# -- NEEDED TO FIND THE REMAINING CONFIGURATION FILES
CFG_BASE=/etc/rdz
CLASSPATH=.:$CFG_BASE:$CLASSPATH
# --
```

3. シェル・スクリプトである `/etc/rdz/update.sh` を作成します。それにより、マスター `rsed.envvars` のコピーと `_RSE_PORTRANGE` の調整を行います。

```
$ oedit /etc/rdz/update.sh
$ chmod 755 /etc/rdz/update.sh
```

```

#!/bin/sh
# Licensed materials - Property of IBM
# 5724-T07 Copyright IBM Corp. 2012
# clone rsed.envvars and set PORTRANGE for use with RDz & DDVIPA

file=rsed.envvars           #; echo file $file
sys=${1:-$(sysvar SYSNAME)} #; echo sys $sys
dir=$(dirname $0)           #; echo dir $dir
# if sysname has a special char, precede it with \ (eg. SYS\1)
case "$sys" in
    "SYS1") range=8108-8118;;
    "SYS2") range=8119-8129;;
esac                         #; echo range $range
echo "setting port range $range for $sys using $dir/$file"

if test ! $range ; then
    echo ERROR: no port range defined for $sys ; exit 12 ; fi
if test ! -e $dir/$file ; then
    echo ERROR: file $dir/$file does not exist ; exit 12 ; fi
if test ! -d $dir/$sys ; then
    echo ERROR: directory $dir/$sys does not exist ; exit 12 ; fi

mv $dir/$sys/$file $dir/$sys/prev.$file 2>/dev/null
sed="/_RSE_PORTRANGE/s/.*/_RSE_PORTRANGE=$range/"
sed "$sed" $dir/$file > $dir/$sys/$file

if test ! -s $dir/$sys/$file ; then
    echo ERROR creating $dir/$sys/$file, restoring backup
    mv $dir/$sys/prev.$file $dir/$sys/$file ; exit 8 ; fi

```

図 11. *update.sh* - ファイアウォールによる *DDVIPA* セットアップのサポート

4. */etc/rdz/SYS1* と */etc/rdz/SYS2* のディレクトリーを作成し、*/etc/rdz/update.sh* を実行して、そのディレクトリーに追加します。

```

$ mkdir /etc/rdz/SYS1 /etc/rdz/SYS2
$ /etc/rdz/update.sh SYS1
setting port range 8108-8118 for SYS1 using
/etc/rdz/rsed.envvars
$ /etc/rdz/update.sh SYS2
setting port range 8119-8129 for SYS2 using
/etc/rdz/rsed.envvars

```

5. *RSED* 開始タスクが、必ず、*/etc/rdz/&SYSNAME* をポイントするようにします。

```
//      CNFG='/etc/rdz/&SYSNAME.'
```

次に、シスプレックス内のポート番号を固有のまま保持するために、定義されたポート範囲がシスプレックス内のすべてのシステム上で *Developer for System z* に予約されていることを確認してください。それぞれのシステムですべての範囲を予約するには、*PROFILE.TCPIP* 内の *PORT* あるいは *PORTRANGE* ステートメントを使用してください。 *RSE* スレッド・プールのジョブ名は、*RSEdx* です。ここで、*RSED* は *RSE* 開始タスクの名前で、*x* はランダムな 1 桁の数値です。したがって、定義内にワイルドカードが必要です。

```

PORTRange 8108 22 RSED*           ; 8108-8129 - Developer for System z
                                   ; - secondary connection

```

8 ページの『接続のフロー』で記されているように、*_RSE_PORTRANGE* 内のポート範囲は小さい可能性があります。 *RSE* サーバーはクライアント接続の期間中に、ポートを独占的に必要とすることはありません。他の *RSE* サーバーがそのポートにバインドできないのは、(サーバーの) バインドから (クライアントの) 接続まで

の間だけです。つまり、ほとんどの接続が範囲内の最初のポートを使用し、同じ範囲内の残りのポートは、同時に複数のログオンが発生したときのバッファになる、ということです。

サンプル・セットアップ

以下のサンプル・セットアップには、SYS1 と SYS2 という、2 つの z/OS システムがあり、これらはシスプレックスの一部となっています。システム SYS1 は、通常、Developer for System z の Distributed DVIPA 用のシスプレックス・ディストリビューターをホストするシステムとして定義されます。

Distributed DVIPA を定義した後、Developer for System z をシステムで始動して、システム間でロード・バランシング・クライアント接続を行うことができます。JES ジョブ・モニターは、ローカル RSE としか対話しないため、DVIPA のセットアップを必要としません。クライアントは、IP アドレス 10.10.10.1 のポート 4035 に接続します。

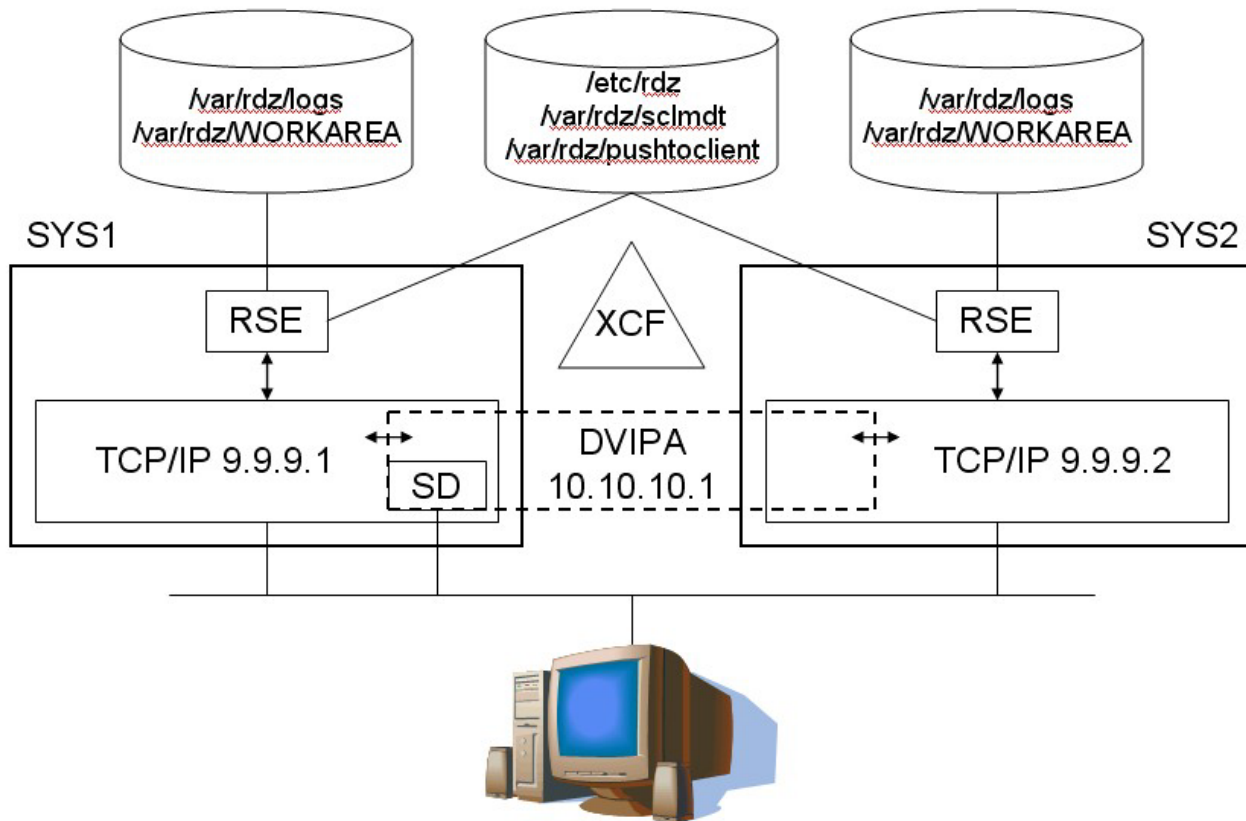


図 12. Distributed Dynamic VIPA サンプル

システム SYS1 - TCP/IP プロファイル

```
IPCONFIG
  SYSPLEXRouting
; SYSPLEXROUTING is required as this stack needs sysplex communication
  DYNAMICXCF 9.9.9.1 255.255.255.0 1
; DYNAMICXCF defines device/link with home address 9.9.9.1 as needed
```

IGNORERedirect

VIPADYNAMIC

VIPADefine 255.255.255.0 10.10.10.1

; VIPADefine defines 10.10.10.1 as main DVIPA on SYS1 for RDz

VIPADISTRIBUTE DEFINE

; VIPADISTRIBUTE makes 10.10.10.1 a distributed DVIPA, must match SYS2

SYSplexExports ; RDz prereq

DISTMethod BASEWLM ; BASEWLM or ROUNDROBIN

10.10.10.1 ; DVIPA address used by RDz clients

PORT 4035 ; port used by RDz clients

DESTIP 9.9.9.1 9.9.9.2 ; RDz active on SYS1 and SYS2

ENDVIPADYNAMIC

システム SYS2 - TCP/IP プロファイル

IPCONFIG

SYSplexRouting

; SYSplexROUTING is required as this stack needs sysplex communication

DYNAMICXCF 9.9.9.2 255.255.255.0 1

; DYNAMICXCF defines device/link with home address 9.9.9.2 as needed

IGNORERedirect

VIPADYNAMIC

VIPABACKUP 255.255.255.0 10.10.10.1

; VIPABACKUP defines 10.10.10.1 as backup DVIPA on SYS2 for RDz

VIPADISTRIBUTE DEFINE

; VIPADISTRIBUTE makes 10.10.10.1 a distributed DVIPA, must match SYS1

SYSplexExports ; RDz prereq

DISTMethod BASEWLM ; BASEWLM or ROUNDROBIN

10.10.10.1 ; DVIPA address used by RDz clients

PORT 4035 ; port used by RDz clients

DESTIP 9.9.9.1 9.9.9.2 ; RDz active on SYS1 and SYS2

ENDVIPADYNAMIC

第 4 章 WLM に関する考慮事項

従来の z/OS アプリケーションとは異なり、Developer for System z は、ワークロード・マネージャー (WLM) で容易に識別できる一体構造のアプリケーションではありません。Developer for System z は、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。3 ページの『第 1 章 Developer for System z について』で説明しているように、これらのサービスの一部は異なるアドレス・スペースでアクティブとなるため、WLM 分類も異なることになります。

この章では、以下のトピックについて説明します。

- 『ワークロード分類』
- 79 ページの『目標の設定』

ワークロード分類

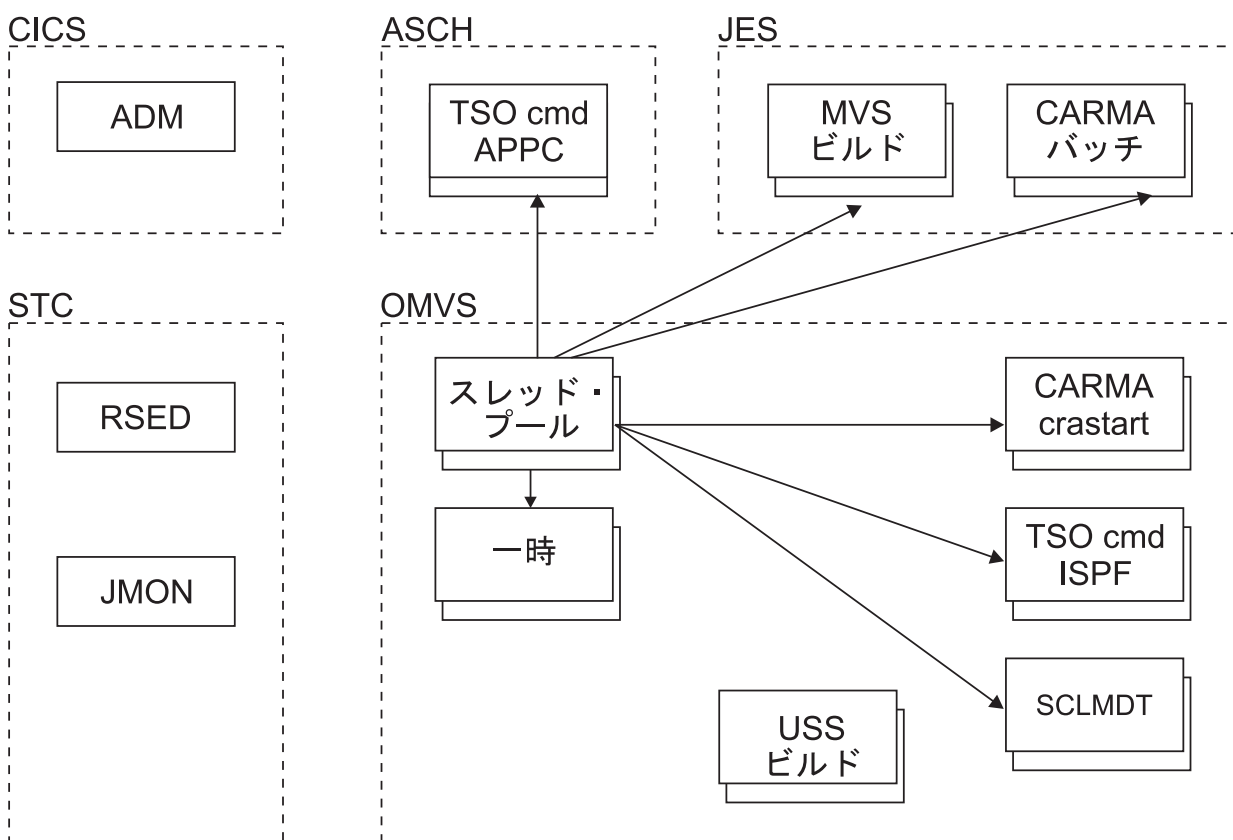


図 13. WLM 分類

図 13 は、Developer for System z ワークロードが WLM に提示されるときに経由するサブシステムの基本的概要を示しています。

Application Deployment Manager (ADM) は CICS 領域でアクティブになるため、WLM での CICS 分類規則に従います。

RSE デーモン (RSED) および JES ジョブ・モニター (JMON) は、Developer for System z の開始タスク (または長期実行バッチ・ジョブ) であり、それぞれが専用のアドレス・スペースを使用します。

5 ページの『Java アプリケーションとしての RSE』で説明しているように、RSE デーモンは RSE スレッド・プール・サーバー (不定数のクライアントをサポート) ごとに子プロセスを spawn します。各スレッド・プールは個別のアドレス・スペースでアクティブになります (z/OS UNIX イニシエーター BPXAS を使用)。これらは spawn されたプロセスであるため、開始タスクの分類規則ではなく、WLM OMVS の分類規則を使用して分類されます。

ユーザーが実行するアクションによっては、スレッド・プール内でアクティブなクライアントが他のアドレス・スペースを多数作成する可能性があります。Developer for System z の構成によっては、TSO コマンド・サービス (TSO cmd) や CARMA などの一部のワークロードが、異なるサブシステムで実行される可能性があります。

77 ページの図 13 に示されているアドレス・スペースは、表示の対象となるほど長くシステムに残りますが、z/OS UNIX の設計仕様のために、存続期間の短い一時的なアドレス・スペースもいくつか存在することに注意してください。これらの一時的なアドレス・スペースは、OMVS サブシステム内でアクティブになります。

RSE スレッド・プールは RSE デーモンと同じユーザー ID および同様のジョブ名を使用しますが、スレッド・プールによって開始されるアドレス・スペースはどれも、アクションを要求しているクライアントのユーザー ID によって所有されることに注意してください。このクライアント・ユーザー ID は、スレッド・プールによって開始されるすべての OMVS ベース・アドレス・スペースのジョブ名 (の一部) としても使用されます。

Developer for System z が使用するその他のサービス (File Manager (FMNCAS) や z/OS UNIX REXEC (USS ビルド) など) によって、さらにアドレス・スペースが作成されます。

分類規則

WLM は、分類規則を使用して、システムに入ってきた作業をサービス・クラスにマッピングします。この分類は、作業修飾子に基づいています。最初の (必須) 修飾子は、作業要求を受け取るサブシステム・タイプです。表 13 に、Developer for System z ワークロードを受け取る可能性があるサブシステム・タイプを示します。

表 13. WLM エントリー・ポイント・サブシステム

サブシステム・タイプ	作業の説明
ASCH	作業要求には、IBM 提供の APPC/MVS トランザクション・スケジューラー ASCH によってスケジュールされるすべての APPC トランザクション・プログラムが含まれます。

表 13. WLM エントリー・ポイント・サブシステム (続き)

サブシステム・タイプ	作業の説明
CICS	作業要求には、CICS によって処理されるすべてのトランザクションが含まれます。
JES	作業要求には、JES2 または JES3 が開始するすべてのジョブが含まれます。
OMVS	作業要求には、z/OS UNIX システム・サービスで fork された子のアドレス・スペースで処理される作業が含まれます。
STC	作業要求には、START および MOUNT コマンドによって開始されるすべての作業が含まれます。STC には、システム・コンポーネント・アドレス・スペースも含まれます。

表 14 に、ワークロードを特定のサービス・クラスに割り当てるために使用できる追加の修飾子を示します。リストされている作業修飾子の詳細については、「MVS 計画: ワークロード管理」(SA88-8574) を参照してください。

表 14. WLM 作業修飾子

		ASCH	CICS	JES	OMVS	STC
AI	アカウント情報	x		x	x	x
LU	LU 名 (*)		x			
PF	実行 (*)			x		x
PRI	優先順位			x		
SE	スケジューリング環境名			x		
SSC	サブシステム・コレクション名			x		
SI	サブシステム・インスタンス (*)		x	x		
SPM	サブシステム・パラメーター					x
PX	シスプレックス名	x	x	x	x	x
SY	システム名 (*)	x			x	x
TC	トランザクション/ジョブ・クラス (*)	x		x		
TN	トランザクション/ジョブ名 (*)	x	x	x	x	x
UI	ユーザー ID (*)	x	x	x	x	x

注: (*) のマークが付いた修飾子については、タイプの省略形に G を付加することで、分類グループを指定できます。例えば、トランザクション名グループは TNG となります。

目標の設定

77 ページの『ワークロード分類』で説明しているように、Developer for System z はシステム上でさまざまなタイプのワークロードを作成します。これらの各種タスクは互いに通信します。つまり、タスク間接続でのタイムアウトの問題を回避するためには、実際の経過時間が重要になります。このため、Developer for System z の

タスクは、ハイパフォーマンスのサービス・クラスに配置するか、または優先順位の高い適度なパフォーマンスのサービス・クラスに配置する必要があります。

したがって、現行の WLM の目標を改訂または更新することをお勧めします。これは特に、従来の MVS 作業現場で時間依存型の OMVS ワークロードを初めて扱う場合に当てはまります。

注:

- このセクションに記載する目標の情報は、あえて説明レベルにとどめています。これは、実際のパフォーマンス目標がサイトによって大きく異なるためです。
- システムでの特定のタスクの影響を理解しやすくするために、最少のリソース使用量、中程度のリソース使用量、および相当なリソース使用量といった言葉を使用しています。これらはいずれも、システム全体ではなく Developer for System z の総リソース使用量を基準としています。

表 15 は、Developer for System z が使用するアドレス・スペースを示しています。z/OS UNIX では、「タスク名」列の「x」がランダムな 1 桁の数値で置き換えられます。

表 15. WLM ワークロード

説明	タスク名	ワークロード
JES ジョブ・モニター	JMON	STC
RSE デーモン	RSED	STC
RSE スレッド・プール	RSEDx	OMVS
ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)	<userid>x	OMVS
TSO コマンド・サービス (APPC)	FEKFRSRV	ASCH
CARMA (パッチ)	CRA<port>	JES
CARMA (crastart)	<userid>x	OMVS
CARMA (ISPF クライアント・ゲートウェイ)	<userid> および <userid>x	OMVS
MVS ビルド (パッチ・ジョブ)	*	JES
z/OS UNIX ビルド (シェル・コマンド)	<userid>x	OMVS
z/OS UNIX シェル	<userid>	OMVS
File Manager タスク	<userid>x	OMVS
Application Deployment Manager	CICSTS	CICS

目標の選択に関する考慮事項

以下に示す WLM の一般的な考慮事項は、Developer for System z に対して適切に目標を定義するために役立ちます。

- 望ましい結果ではなく、実際に達成できることに基づいて目標を設定してください。目標を必要以上に高く設定すると、WLM は重要性の低い作業から重要性の高い作業にリソースを移しますが、この重要性の高い作業が実際にはリソースを必要としない場合があります。

- **SYSTEM** および **SYSSTC** サービス・クラスに割り当てられる作業量を制限してください。これは、これらのクラスのディスパッチング優先順位が **WLM** 管理クラスよりも高いためです。この 2 つのクラスは、重要性が高く、かつ **CPU** の使用量が少ない作業に使用してください。
- 分類規則の対象から外れた作業は、任意の目標を持つ **SYSOTHER** クラスに割り当てられることになります。任意の目標では、**WLM** に対してシステムに予備のリソースがあるときに最善策を取ることだけが指示されます。

応答時間目標を使用する場合:

- **WLM** で応答時間目標を適切に管理するには、タスクの到着率が一定であることが必要です (少なくとも 20 分間に 10 タスク)。
- 平均応答時間の目標は、十分に制御されたワークロードにのみ使用してください。1 つの長いトランザクションが平均応答時間に大きく影響し、**WLM** が過剰反応する可能性があるからです。

速度目標を使用する場合:

- さまざまな理由から、通常は速度目標の達成率が 90% を超えることは不可能です。例えば、**SYSTEM** および **SYSSTC** アドレス・スペースのディスパッチング優先順位は、すべて速度タイプの目標を上回っています。
- **WLM** は、その速度目標を、最小限の数の (使用および遅延) サンプルに基づいて決定します。このため、サービス・クラスで実行されている作業が少ないほど、必要な数のサンプルの収集とディスパッチング・ポリシーの調整に時間がかかることになります。
- ハードウェアを変更する場合は、速度目標を再評価してください。特に、より少ない台数のより高速なプロセッサへと移行する場合は、速度目標の変更が必要となります。

STC

Developer for System z の開始タスクである **RSE** デーモン および **JES** ジョブ・モニターはいずれも、リアルタイムのクライアント要求を処理します。

表 16. **WLM** ワークロード - **STC**

説明	タスク名	ワークロード
JES ジョブ・モニター	JMON	STC
RSE デーモン	RSED	STC

- **JES** ジョブ・モニター

JES ジョブ・モニターは、ジョブの実行依頼、スプール・ファイルの表示、**JES** オペレーター・コマンドの実行など、すべての **JES** 関連サービスを提供します。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが **WLM** に個々のトランザクションを報告しないためです。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少から中程度と予想されます。

- **RSE** デーモン

RSE デーモンは、クライアントのログオンと認証を処理し、複数の RSE スレッド・プールを管理します。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが WLM に個々のトランザクションを報告しないためです。リソース使用量は、作業日の開始点をピークとして、中程度と予想されます。

OMVS

OMVS ワークロードは 2 つのグループ、つまり RSE スレッド・プールとそれ以外に分けることができます。これは、RSE スレッド・プールを除くすべてのワークロードが、クライアント・ユーザー ID をアドレス・スペース名のベースとして使用するためです (z/OS UNIX では、「タスク名」列の「x」がランダムな 1 桁の数値で置き換えられます)。

表 17. WLM ワークロード - OMVS

説明	タスク名	ワークロード
RSE スレッド・プール	RSEDx	OMVS
ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)	<userid>x	OMVS
CARMA (crastart)	<userid>x	OMVS
CARMA (ISPF クライアント・ゲートウェイ)	<userid> および <userid>x	OMVS
z/OS UNIX ビルド (シェル・コマンド)	<userid>x	OMVS
z/OS UNIX シェル	<userid>	OMVS
File Manager タスク	<userid>x	OMVS

• RSE スレッド・プール

RSE スレッド・プールは、Developer for System z の中枢であると言えます。ほぼすべてのデータがスレッド・プールを通過し、スレッド・プール内のマイナー (ユーザー固有のスレッド) によって、他の Developer for System z 関連タスクのほとんどが制御されます。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが WLM に個々のトランザクションを報告しないためです。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、相当量と予想されます。

残りのワークロードは、アドレス・スペース命名規則が共通であるために、すべて同じサービス・クラスに割り当てられることになります。このサービス・クラスには、複数期間の目標を指定する必要があります。最初の期間にはハイパフォーマンスのパークンタイル応答時間目標を指定し、最後の期間には適度なパフォーマンスの速度目標を指定する必要があります。ISPF クライアント・ゲートウェイなどの一部のワークロードは、個々のトランザクションを WLM に報告しますが、それ以外のワークロードはこれを行いません。

• ISPF クライアント・ゲートウェイ

ISPF クライアント・ゲートウェイは、Developer for System z が非対話式の TSO コマンドと ISPF コマンドを実行するために呼び出す ISPF サービスです。これ

には、クライアントが発行する明示的なコマンドと、Developer for System z が発行する暗黙的なコマンド (PDS メンバー・リストの取得など) が含まれます。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

- CARMA

CARMA はオプションの Developer for System z サーバーで、CA Endeavor® SCM などのホスト・ベースの Software Configuration Manager (SCM) と対話するために使用されます。Developer for System z では、CARMA サーバーをさまざまな方式で始動することができ、その一部は OMVS ワークロードになります。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

- z/OS UNIX ビルド

クライアントが z/OS UNIX プロジェクトのビルドを開始すると、z/OS UNIX REXEC (または SSH) によって、ビルドを実行するための多数の z/OS UNIX シェル・コマンドを実行するタスクが開始されます。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、プロジェクトのサイズに応じて中程度から相当量と予想されます。

- z/OS UNIX シェル

このワークロードは、クライアントによって発行される z/OS UNIX シェル・コマンドを処理します。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

- IBM File Manager

spawn された File Manager 子プロセスは、Developer for System z アドレス・スペースではありませんが、Developer for System z クライアントの要求に応じて開始でき、これらのタスクが Developer for System z タスクと同じ命名規則を使用することから、ここにリストしています。これらの File Manager タスクは、重要な MVS データ・セット・アクション (VSAM ファイルのフォーマット編集など) を処理します。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少から中程度と予想されます。

JES

JES で管理されるバッチ処理は、Developer for System z によってさまざまに使用されます。最も一般的な用途は、MVS ビルドです。ここでは、ジョブが実行依頼され、終了のタイミングを判別するためにモニターされます。ただし、Developer for System z は、CARMA サーバーをバッチで始動し、TCP/IP を使用してそのサーバーと通信することもできます。

表 18. WLM ワークロード - JES

説明	タスク名	ワークロード
CARMA (バッチ)	CRA<port>	JES
MVS ビルド (バッチ・ジョブ)	*	JES

- CARMA

CARMA はオプションの Developer for System z サーバーで、CA Endevor® SCM などのホスト・ベースの Software Configuration Manager (SCM) と対話するために使用されます。Developer for System z では、CARMA サーバーをさまざまな方法で始動することができ、その一部は JES ワークロードになります。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが WLM に個々のトランザクションを報告しないためです。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

- MVS ビルド

クライアントが MVS プロジェクトのビルドを開始すると、Developer for System z によって、ビルドを実行するためのバッチ・ジョブ開始されます。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、プロジェクトのサイズに応じて中程度から相当量と予想されます。ご使用のローカル環境に応じて、適度のパフォーマンスを目標とするさまざまな戦略をお勧めできます。

- パーセンタイル応答時間目標の期間と、後続の速度目標の期間で構成される、複数期間の目標を指定できます。この場合、開発者は、応答時間が均一のジョブを作成するために、ほぼ同じビルド・プロシージャとほぼ同じサイズの入力ファイルを使用する必要があります。WLM で応答時間目標を適切に管理するには、ジョブの到着率が一定であることが必要です (少なくとも 20 分間に 10 ジョブ)。
- 速度目標は、実行時間と到着率にかなりのばらつきがあっても対応できるため、ほとんどのバッチ・ジョブに適しています。

ASCH

現行の Developer for System z バージョンでは、ISPF クライアント・ゲートウェイを使用して、非対話式の TSO コマンドと ISPF コマンドが実行されます。歴史的な理由から、Developer for System z は APPC トランザクションによるこれらのコマンドの実行もサポートしています。APPC 方式は推奨されないことに注意してください。

表 19. WLM ワークロード - ASCH

説明	タスク名	ワークロード
TSO コマンド・サービス (APPC)	FEKFRSRV	ASCH

- TSO コマンド・サービス

TSO コマンド・サービスは、非対話式の TSO コマンドと ISPF コマンドを実行するために、Developer for System z によって APPC トランザクション・プログラムとして開始することができます。これには、クライアントが発行する明示的なコマンドと、Developer for System z が発行する暗黙的なコマンド (PDS メンバー・リストの取得など) が含まれます。このサービス・クラスには、複数期間の目標を指定する必要があります。最初の期間には、ハイパフォーマンスのパーセンタイル応答時間目標を指定してください。最後の期間には適度なパフォーマンスの速度目標を指定してください。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

CICS

Application Deployment Manager は、CICS Transaction Server 領域内でアクティブになるオプションの Developer for System z サーバーです。

表 20. WLM ワークロード - CICS

説明	タスク名	ワークロード
Application Deployment Manager	CICSTS	CICS

- Application Deployment Manager

CICSTS 領域でアクティブになるオプションの Application Deployment Manager サーバーでは、選択した CICSTS 管理タスクを開発者に安全にオフロードすることができます。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。使用すべきサービス・クラスのタイプは、当該 CICS 領域内でアクティブな他のトランザクションによって異なるため、ここでは詳しく説明しません。

WLM は、CICS に使用できる複数の管理タイプをサポートしています。

- 領域目標に向けた CICS の管理

CICS アドレス・スペースを管理するサービス・クラスに目標が設定されます。ユーザーは、このサービス・クラスの実行速度目標のみを使用できます。WLM は、アドレス・スペースに対する JES または STC の分類規則を使用しますが、トランザクションに対する CICS サブシステムの分類規則は使用しません。

- トランザクション応答時間目標に向けた CICS の管理

単一のトランザクションまたはトランザクションのグループに割り当てられたサービス・クラスにおいて、応答時間目標を設定することができます。WLM は、アドレス・スペースに対する JES または STC の分類規則、およびトランザクションに対する CICS サブシステムの分類規則を使用します。

第 5 章 チューニングに関する考慮事項

3 ページの『第 1 章 Developer for System z について』で説明されているように、RSE (リモート・システム・エクスプローラー) は、Developer for System z の中核です。クライアントからの接続とワークロードを管理するために、RSE は、スレッド・プーリング・アドレス・スペースを制御するデーモン・アドレス・スペースから構成されています。デーモンは接続と管理のためのフォーカル・ポイントとして機能し、スレッド・プールはクライアント・ワークロードを処理します。

このため、RSE は Developer for System z セットアップをチューニングする場合の主要な対象となります。ただし、それぞれが 17 個以上のスレッドを使用する何百人ものユーザー、ある程度の大きさのストレージ、そして場合によっては 1 つ以上のアドレス・スペースを保守するには、Developer for System z と z/OS の両方を適切に構成する必要があります。

この章では、以下のトピックについて説明します。

- 『リソース使用量』
- 99 ページの『ストレージの使用量』
- 105 ページの『z/OS UNIX ファイル・システム・スペースの使用量』
- 108 ページの『主要なリソース定義』
- 112 ページの『さまざまなリソース定義』
- 114 ページの『モニター』
- 118 ページの『サンプル・セットアップ』

リソース使用量

このセクションの情報を使用して、Developer for System z の標準のリソース使用量と最大のリソース使用量を見積もり、それに合わせてシステム構成を計画することができます。

このセクションで示す数値と数式を使用してシステム限度を定義するときには、使用する見積もり値がきわめて正確であることに留意してください。システム限度を設定するときは、一時タスクやその他のタスク、あるいは同じホストに同時に複数回接続する (例えば RSE と TN3270 経由で) ユーザーがリソースを使用できるように、十分なゆとりを残すようにします。

注:

- ここで示す情報の範囲は、Developer for System z 自体が提供し RSE 経由でアクセスされるサービスに限定されます。例えば、TN3270 のリソース使用量 (RSE 経由でアクセスされません) や、MVS または z/OS UNIX プロジェクトのリモート (ホスト・ベースの) ビルドで呼び出されるプログラムのリソース使用量 (Developer for System z から提供されません) については記載していません。
- Developer for System z にサード・パーティーの拡張を追加すると、リソース使用量のカウンターが増えることがあります。

- どのサービスにも短時間で終了する「ハウスキーピング」タスクがあります。これらは、その実行中にリソースを使用し、互いに順次または並列に実行されます。これらのタスクが使用するリソースについては記載していません。
- ISPF クライアント・ゲートウェイなどの必要なソフトウェアによるユーザー固有のリソース使用量については、有用な情報のみを記載しています。
- ここに示す数値は、事前の通知なしに変更される可能性があります。

概要

以下の各表は、Developer for System z で使用されるアドレス・スペース、プロセス、およびスレッドの数の概要です。ここに示す数値については、次のセクションで詳しく説明します。

- 89 ページの『アドレス・スペースの数』
- 92 ページの『プロセスの数』
- 95 ページの『スレッドの数』

表 21 は、Developer for System z の開始タスクで使用される主要なリソースの概要です。これらのリソースは、1 回だけ割り振られ、すべての Developer for System z クライアント間で共有されます。

表 21. 共通のリソース使用量

開始タスク	アドレス・スペース数	プロセス数	スレッド数
JMON	1	1	3
RSED	1	3	11
RSEDx	1 + 1 (a)	1 + 2	1 + 10

注: (a) 1 つの APF 許可アドレス・スペースと、少なくとも 1 つの RSE スレッド・プール・アドレス・スペースがアクティブです。RSE スレッド・プール・アドレス・スペースの実際の数を確認するには、89 ページの『アドレス・スペースの数』を参照してください。

表 22 は、必要なソフトウェアで使用される主要なリソースの概要です。これらのリソースは、関連機能呼び出す Developer for System z クライアントごとに割り振られます。

表 22. ユーザーごとに必要なリソース使用量

必要なソフトウェア	アドレス・スペース数	プロセス数	スレッド数
ISPF クライアント・ゲートウェイ	1	2	4
APPC	1	1	2

89 ページの表 23 は、各 Developer for System z クライアントで指定の機能を実行するときに使用される主要なリソースの概要です。数値以外の値 (ISPF など) は、表 22 の対応する値を指しています。

表 23. ユーザーごとのリソース使用量

ユーザーのアクション	アドレス・スペース数	プロセス数	スレッド数		
	ユーザー ID	ユーザー ID	ユーザー ID	RSEDx	JMON
ログオン	-	-	-	17	1
アイドル・タイムアウト用のタイマー	-	-	-	1	-
検索	-	-	-	1	-
PDS(E) の拡張	ISPF	ISPF	ISPF	-	-
データ・セットのオープン	ISPF	ISPF	ISPF	1	-
TSO コマンド	ISPF	ISPF	ISPF	-	-
z/OS UNIX シェル	1	1	1	6	-
MVS ビルド	1	-	-	-	-
z/OS UNIX ビルド	3	3	3	-	-
CARMA (バッチ)	1	1	2	1	-
CARMA (crastart)	1	1	2	4	-
CARMA (ispf)	4	4	7	5	-
SCLMDT	ISPF	ISPF	ISPF	-	-

注: SCLM Developer Toolkit 以外は、ISPF を APPC に置き換えることもできます。

アドレス・スペースの数

表 24 は、Developer for System z で使用されるアドレス・スペース数を示しています。この表の「数」列の「u」は、その機能を使用する同時にアクティブなユーザーの数で値を乗算する必要があることを示します。「タスク名」列の「x」は、z/OS UNIX によってランダムな 1 桁の数値で置き換えられます。

表 24. アドレス・スペースの数

数	説明	タスク名	共用	終了のタイミング
1	JES ジョブ・モニター	JMON	はい	なし
1	RSE デーモン	RSED	はい	なし
1	RSE APF 許可済み	RSEDx	はい	なし
(a)	RSE スレッド・プール	RSEDx	はい	なし
1u	ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)	<userid>x	いいえ	15 分後またはユーザー・ログオフ後

表 24. アドレス・スペースの数 (続き)

数	説明	タスク名	共用	終了のタイミング
1u	TSO コマンド・サービス (APPC)	FEKFRSRV	いいえ	60 分後またはユーザー・ログオフ後
1u	CARMA (バッチ)	CRA<port>	いいえ	7 分後またはユーザー・ログオフ後
1u	CARMA (crastart)	<userid>x	いいえ	7 分後またはユーザー・ログオフ後
4u	CARMA (ispf、非推奨)	(1)<userid> または (3)<userid>x	いいえ	7 分後またはユーザー・ログオフ後
(b)	1 人のユーザーによる ISPF クライアント・ゲートウェイの同時使用	<userid>x	いいえ	タスク完了後
1u	MVS ビルド (バッチ・ジョブ)	*	いいえ	タスク完了後
3u	z/OS UNIX ビルド (シェル・コマンド)	<userid>x	いいえ	タスク完了後
1u	z/OS UNIX シェル	<userid>	いいえ	ユーザー・ログオフ後

注:

- (a) 少なくとも 1 つの RSE スレッド・プール・アドレス・スペースがアクティブです。実際数は以下の要素に依存します。
 - rsed.envvars 内の minimum.threadpool.process ディレクティブ。デフォルト値は 1 です。
 - 1 つのスレッド・プールでサービスできるユーザーの数。デフォルトの設定では、スレッド・プール当たり 30 ユーザーです。

注: single.logon ディレクティブがアクティブな場合、minimum.threadpool.process が 1 に設定されているとしても、少なくとも 2 つのスレッド・プールが開始されます。rsed.envvars 内の single.logon のデフォルトの設定はアクティブです。

- (b) Developer for System z では、1 人のユーザーに対して複数のスレッドがアクティブになります。ISPF クライアント・ゲートウェイのアドレス・スペースが、あるスレッドの要求を処理している間に別のスレッドが要求を送信すると、ISPF は新しいクライアント・ゲートウェイを開始して新しい要求を処理します。このアドレス・スペースは、タスク完了後に終了します。
- SCLMDT は、ISPF クライアント・ゲートウェイ・アドレス・スペースを必要とします。SCLMDT は、このアドレス・スペースを TSO コマンド・サービスと共用します。
- MVS データ・セット関連アクションのほとんどは TSO コマンド・サービスを使用しますが、このサービスは個々に ISPF クライアント・ゲートウェイまたは APPC トランザクションでアクティブとなります。

91 ページの図 14 の数式は、Developer for System z で使用されるアドレス・スペースの最大数を見積もる場合に使用します。

$$3 + A + N*(x + y + z) + (2 + N*0.01)$$

図 14. アドレス・スペースの最大数

各項の説明は次のとおりです。

- 「3」は、永続的にアクティブなサーバー・アドレス・スペースの数です。
- 「A」は、RSE スレッド・プール・アドレス・スペースの数を表します。
- 「N」は、同時ユーザーの最大数を表します。
- 「x」は、選択した構成オプションに応じて、以下のいずれかの値になります。

X	SCLMDT	クライアント・ゲートウェイ経由の TSO	APPC 経由の TSO
1	いいえ	いいえ	はい
1	いいえ	はい	いいえ
1	はい	はい	いいえ

- 「y」は、選択した構成オプションに応じて、以下のいずれかの値になります。

y	
0	CARMA なし
1	CARMA (バッチ)
1	CARMA (crastart)
4	CARMA (ispf、非推奨)

- 「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。
 - MVS ビルドが実行される場合は 1 を加算します。これらのアドレス・スペースは、関連するビルド・タスク (バッチ・ジョブ) が完了した時点で終了します。
 - z/OS UNIX ビルドが実行される場合は 3 を加算します。呼び出されたプログラムの必要性によっては、実際の数値がこれより大きくなる場合があるので注意してください。これらのアドレス・スペースは、関連するビルド・タスクが完了した時点で終了します。
- 「2 + N*0.01」は、一時アドレス・スペース用のバッファを追加します。必要なバッファ・サイズは、ご使用のサイトによって異なる場合があります。

図 15 の数式は、Developer for System z クライアントで使用されるアドレス・スペースの最大数を見積もる場合に使用します (ここに記載していない一時アドレス・スペースはカウントしていません)。

$$x + y + z$$

図 15. クライアントごとのアドレス・スペース数

各項の説明は次のとおりです。

- 「x」は、選択した構成オプションによって異なります。アドレス・スペースの最大数を計算する数式（91 ページの図 14）を参照してください。
- 「y」は、選択した構成オプションによって異なります。アドレス・スペースの最大数を計算する数式（91 ページの図 14）を参照してください。
- 「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。アドレス・スペースの最大数を計算する数式（91 ページの図 14）を参照してください。

表 25 に示す定義によって、アドレス・スペースの実際の数进行制限することができます。

表 25. アドレス・スペースの限度

ロケーショ ン	限度	影響を受けるリソース
rsed.envvars	maximum.threadpool.process	RSE スレッド・プールの数を制限します。
IEASYMxx	MAXUSER	アドレス・スペースの数を制限します。
ASCHPMxx	MAX	TSO コマンド・サービス (APPC) での APPC イニシエーターの数を制限します。

プロセスの数

表 26 は、Developer for System z が使用するアドレス・スペースごとのプロセスの数をリストしています。「アドレス・スペース」列の「u」は、その機能を使用する同時にアクティブなユーザーの数で値を乗算する必要があることを示します。

表 26. プロセスの数

プロセス数	アドレス・ス ペース数	説明	ユーザー ID
1	1	JES ジョブ・モニター	STCJMON
3	1	RSE デーモン	STCRSE
1	1	RSE APF 許可	STCRSE
2	(a)	RSE スレッド・プール	STCRSE
2	(b)	ISPF クライアント・ゲートウェイ (TSO コマ ンド・サービスおよび SCLMDT)	<userid>
1	1u	TSO コマンド・サービス (APPC)	<userid>
1	1u	CARMA (パッチ)	<userid>
1	1u	CARMA (crastart)	<userid>
1	1u	CARMA (ispf、非推奨)	<userid>
1	3u	z/OS UNIX ビルド (シェル・コマンド)	<userid>
1	1u	z/OS UNIX シェル	<userid>
(5)	(u)	SCLM Developer Toolkit	<userid>

注:

- (a) 少なくとも 1 つの RSE スレッド・プール・アドレス・スペースがアクティ
ブです。RSE スレッド・プール・アドレス・スペースの実際の数を確認するに
は、89 ページの『アドレス・スペースの数』を参照してください。

- RSE デーモンおよびすべての RSE スレッド・プールは、同じユーザー ID を使用します。
- (b) 通常の状態デフォルトの構成オプションを使用しているときは、ユーザーごとに 1 つの ISPF クライアント・ゲートウェイがアクティブになります。実際の数には変わる可能性があります (89 ページの『アドレス・スペースの数』を参照してください)。
- SCLMDT は、ISPF クライアント・ゲートウェイ・アドレス・スペースを必要とします。SCLMDT は、このアドレス・スペースを TSO コマンド・サービスと共有します。
- (u) SCLMDT プロセスは ISPF クライアント・ゲートウェイのアドレス・スペースで実行されるため、アドレス・スペース数の値はありません。
- SCLMDT プロセスは一時的であり、タスクが完了すると終了しますが、1 人のユーザーに対して同時に複数のプロセスがアクティブになることがあります。92 ページの表 26 には、同時 SCLMDT プロセスの最大数を示しています。
- MVS データ・セット関連アクションのほとんどは TSO コマンド・サービスを使用しますが、このサービスは個々に ISPF クライアント・ゲートウェイまたは APPC トランザクションでアクティブとなります。
- z/OS UNIX ビルドは合計で 3 つのプロセスを使用し、それぞれが専用のアドレス・スペースで実行されます。
- 特に断りがない限り、リストされているすべてのプロセスは、関連するアドレス・スペースが終了するまでアクティブです。

図 16 の数式は、Developer for System z で使用されるプロセスの最大数を見積もる場合に使用します。

$$5 + 2 * A + N * (x + y + z) + (10 + N * 0.05)$$

図 16. プロセスの最大数

各項の説明は次のとおりです。

- 「5」は、永続的にアクティブなサーバー・アドレス・スペースで使用されるプロセスの数です。
- 「A」は、RSE スレッド・プール・アドレス・スペースの数を表します。
- 「N」は、同時ユーザーの最大数を表します。
- 「x」は、選択した構成オプションに応じて、以下のいずれかの値になります。

X	SCLMDT	クライアント・ゲートウェイ経由の TSO	APPC 経由の TSO
1	いいえ	いいえ	はい
2	いいえ	はい	いいえ
7	はい	はい	いいえ

- 「y」は、選択した構成オプションに応じて、以下のいずれかの値になります。

y	
0	CARMA なし
1	CARMA (パッチ)
1	CARMA (crastart)
4	CARMA (ispf、非推奨)

- 「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。
 - z/OS UNIX シェルが開かれる場合は 1 を加算します。このプロセスは、ユーザーがログオフするまでアクティブです。
 - z/OS UNIX ビルドが実行される場合は 3 を加算します。呼び出されたプログラムの必要性によっては、実際の数値がこれより大きくなる場合があるので注意してください。これらのプロセスは、関連するビルド・タスクが完了した時点で終了します。
- 「10 + N*0.05」は、一時プロセス用のバッファを追加します。必要なバッファ・サイズは、ご使用のサイトによって異なる場合があります。

図 17 の数式は、STCRSE、RSED 開始タスクのユーザー ID で使用されるプロセスの最大数を見積もる場合に使用します (ここに記載していない一時プロセスはカウントしていません)。

$$4 + 2 * A$$

図 17. STCRSE のプロセス数

各項の説明は次のとおりです。

- 「4」は、RSE デーモンおよび RSE APF 許可済みアドレス・スペースで使われるプロセスの数です。
- 「A」は、RSE スレッド・プール・アドレス・スペースの数を表します。

図 18 の数式は、Developer for System z クライアントで使われるプロセスの最大数を見積もる場合に使用します (ここに記載していない一時プロセスはカウントしていません)。

$$(x + y + z) + 5 * s$$

図 18. クライアントごとのプロセス数

各項の説明は次のとおりです。

- 「x」は、選択した構成オプションによって異なります。プロセスの最大数を計算する数式 (93 ページの図 16) を参照してください。
- 「y」は、選択した構成オプションによって異なります。プロセスの最大数を計算する数式 (93 ページの図 16) を参照してください。

- 「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。プロセスの最大数を計算する数式（93 ページの図 16）を参照してください。
- 「s」は、SCLM Developer Toolkit が使用される場合は 1、そうでない場合は 0 です。

表 27 に示す定義によって、プロセスの実際の数を制限することができます。

表 27. プロセスの限度

ロケーション	限度	影響を受けるリソース
BPXPRMxx	MAXPROCSYS	プロセスの総数を制限します。
BPXPRMxx	MAXPROCUSER	z/OS UNIX UID ごとのプロセスの数を制限します。

注:

- RSE デーモンと RSE スレッド・プールは、同じユーザー ID を使用します。RSE デーモンは、必要に応じて新しいスレッド・プールを開始するため、このユーザー ID のプロセス数は増大する可能性があります。そのため、MAXPROCUSER を設定してこの増大に対応する必要があります。これは、「3 + 2*A」と数式化することができます。
- MAXPROCUSER 限度は、固有の z/OS UNIX ユーザー ID (UID) ごとに設定します。複数のユーザーが同じ UID を共用する場合は、見積もったユーザー単位のプロセス数に、同時にアクティブなクライアントの数を乗算します。

スレッドの数

表 28 は、選択された Developer for System z の機能で使用するスレッドの数を示しています。「スレッド数」列の「u」は、その機能を使用する同時にアクティブなユーザーの数で値を乗算する必要があることを示します。スレッドの数はプロセス単位で示しています。これは、限度がこのレベルで設定されるためです。

- RSEDx: ここに示すスレッドは、複数のクライアントで共用される RSE スレッド・プールで作成されます。総数を得るには、最終的に同じスレッド・プールに入るすべてのスレッドを合算する必要があります。
- アクティブ: ここに示すスレッドは、要求された機能を実際に行うプロセスの一部です。各プロセスは独立した単位であるため、特に断りがない限り、それらのプロセスが同じユーザー ID に割り当てられていてもスレッド数を合計する必要はありません。
- ブートストラップ: ブートストラップ・プロセスは、実際のプロセスを開始するために必要です。それぞれが 1 つのスレッドを使用し、複数のブートストラップが連続している可能性があります。スレッド数を合計する必要はありません。

表 28. スレッドの数

スレッド数			ユーザー ID	説明
RSEDx	アクティブ	ブートストラップ		
-	3 + 1u	-	STCJMON	JES ジョブ・モニター

表 28. スレッドの数 (続き)

スレッド数			ユーザー ID	説明
-	15	2	STCRSE	RSE デーモン
-	1	-	STCRSE	RSE APF 許可済み
10 (a) + 17u	-	1 (a)	STCRSE	RSE スレッド・プール
-	4u (b)	1u (b)	<userid>	ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)
-	2u	-	<userid>	TSO コマンド・サービス (APPC)
1u	2u	-	STCRSE および <userid>	CARMA (バッチ)
4u	2u	-	STCRSE および <userid>	CARMA (crastart)
5u	4u	3u	STCRSE および <userid>	CARMA (ispf、非推奨)
-	1u (c)	2u	<userid>	z/OS UNIX ビルド (シェル・コマンド)
6u	1u	-	STCRSE および <userid>	z/OS UNIX シェル
1 (d)	-	-	STCRSE	ダウンロード
1 (e)	-	-	STCRSE	検索
-	(5)	-	<userid>	SCLM Developer Toolkit
1u	-	-	STCRSE	アイドル・タイムアウト用のタイマー

注:

- (a) 少なくとも 1 つの RSE スレッド・プール・アドレス・スペースがアクティブです。RSE スレッド・プール・アドレス・スペースの実際の数を確認するには、89 ページの『アドレス・スペースの数』を参照してください。
- (b) 通常の状態デフォルトの構成オプションを使用しているときは、ユーザーごとに 1 つの ISPF クライアント・ゲートウェイがアクティブになります。実際数は変わる可能性があります (89 ページの『アドレス・スペースの数』を参照してください)。
- SCLMDT は、ISPF クライアント・ゲートウェイ・アドレス・スペースを必要とします。SCLMDT は、このアドレス・スペースを TSO コマンド・サービスと共有します。
- 選択されたアクションによっては、SCLMDT がタスク完了時に終了する単一スレッド・プロセスを複数使用する場合があります。95 ページの表 28 には、同時 SCLMDT スレッドの最大数を示しています。

- MVS データ・セット関連アクションのほとんどは TSO コマンド・サービスを使用しますが、このサービスは個々に ISPF クライアント・ゲートウェイまたは APPC トランザクションでアクティブとなります。
- (c) z/OS UNIX ビルドがさまざまなビルド・ユーティリティを呼び出して、ビルドがマルチスレッド化する場合があります。95 ページの表 28 には、同時 z/OS UNIX ビルド・スレッドの最小数を示しています。
- (d) ホスト・データの各ダウンロードには別のスレッドを使用します。このスレッドは、データがクライアントに一度転送されると終了します。
- (e) 各リモート検索には別のスレッドを使用します。このスレッドは、結果がクライアントに一度転送されると終了します。
- 特に断りがない限り、リストされているすべてのスレッドは、関連するプロセスが終了するまでアクティブです。
- RSE APF 許可済みコードの通常のスレッド数は 1 ですが、始動時には一時的に 13 個以上のスレッドが同時にアクティブになります。

図 19 の数式は、RSE スレッド・プールで使用されるスレッドの最大数を見積もる場合に使用します。図 20 の数式は、JES ジョブ・モニターで使用されるスレッドの最大数を見積もる場合に使用します。

$$10 + N * (17 + x + y + z) + (20 + N * 0.1)$$

図 19. RSE スレッド・プール・スレッドの最大数

$$3 + N$$

図 20. JES ジョブ・モニター・スレッドの最大数

各項の説明は次のとおりです。

- 「N」は、このスレッド・プールまたは JES ジョブ・モニターでの同時ユーザーの最大数を表します。デフォルトの設定では、スレッド・プール当たり 30 ユーザーです。
- 「x」は、選択した構成オプションに応じて、以下のいずれかの値になります。

X	SCLMDT	クライアント・ゲート ウェイ経由の TSO	APPC 経由の TSO	タイムアウト
0	いいえ	いいえ	はい	いいえ
0	いいえ	はい	いいえ	いいえ
0	はい	はい	いいえ	いいえ
1	いいえ	いいえ	はい	はい
1	いいえ	はい	いいえ	はい
1	はい	はい	いいえ	はい

- 「y」は、選択した構成オプションに応じて、以下のいずれかの値になります。

y	
0	CARMA なし
1	CARMA (バッチ)
4	CARMA (crastart)
5	CARMA (ispf、非推奨)

- 「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。
 - z/OS UNIX シェルが開かれる場合は 6 を加算します。これらのスレッドは、ユーザーがログオフするまでアクティブです。
- 「20 + N*0.1」は、一時スレッド用のバッファを追加します。必要なバッファ・サイズは、ご使用のサイトによって異なる場合があります。同時に複数のダウンロードおよび検索をすることは、このバッファ・サイズを増やす必要がある 2 つの例です。

表 29 に示す定義によって、プロセス内の実際のスレッド数を制限することができません。これは、主に RSE スレッド・プールにとって重要です。

表 29. スレッドの限度

ロケーション	限度	影響を受けるリソース
BPXPRMxx	MAXTHREADS	プロセス内のスレッドの数を制限します。
BPXPRMxx	MAXTHREADTASKS	プロセス内の MVS タスクの数を制限します。
BPXPRMxx	MAXASSIZE	アドレス・スペース・サイズを制限し、それによってスレッドに関連する制御ブロックで使用可能なストレージの大きさを制限します。
rsed.envvars	Xmx	Java ヒープ・サイズの最大値を設定します。このストレージは予約され、スレッドに関連する制御ブロックには使用できなくなります。
rsed.envvars	maximum.clients	RSE スレッド・プール内のクライアント (およびそのスレッド) の数を制限します。
rsed.envvars	maximum.threads	RSE スレッド・プール内のクライアント・スレッドの数を制限します。
FEJCNFG	MAX_THREADS	JES ジョブ・モニターでのスレッドの数を制限します。

注:

- rsed.envvars 内の maximum.threads の値は、BPXPRMxx 内の MAXTHREADS および MAXTHREADTASKS の値よりも小さくする必要があります。
- 「**DISPLAY PROCESS,CPU**」オペレーター・コマンドは、スレッド・プール内のアクティブなスレッドを表示しますが、最初の 4000 スレッドのみ表示されるように制限されています。

一時的なリソース使用量

この前までのセクションで説明したリソース使用量は、Developer for System z の存続期間に対して永続的、または特定のユーザー固有のタスクに対して半永続的です。

しかし、Developer for System z は、ハウスキーピング・タスクのため、および以下の要求を満たすために、追加のリソースを一時的に使用します。

- 監査ファイル・イベントの処理 (rsed.envvars 内の audit.action ディレクティブ) では、追加の 1 つのスレッドと、追加の 1 つのプロセスが使用されます。また (audit.action.id がセットされた場合に)、追加の 1 つのアドレス・スペースを使用することがあります。
- ログオン・イベントの処理 (rsed.envvars 内の logon.action ディレクティブ) では、追加の 1 つのスレッドと、追加の 1 つのプロセスが使用されます。また (logon.action.id がセットされた場合に)、追加の 1 つのアドレス・スペースを使用することがあります。
- オペレーター・コマンド IVP PASSTICKET では、追加の 2 つのスレッドを使用します。
- オペレーター・コマンド IVP DAEMON では、追加の 1 つのスレッドと、追加の 1 つのプロセスと、追加の 1 つのアドレス・スペースを使用します。
- オペレーター・コマンド IVP ISPF では、追加の 1 つのスレッドと、追加の 1 つのプロセスと、追加の 1 つのアドレス・スペースを使用するほか、ISPF クライアント・ゲートウェイの使用するリソースが追加で消費されます。

ストレージの使用量

RSE は Java アプリケーションであるため、Developer for System z でのストレージ (メモリー) 使用量を計画する際には、ストレージの割り振りに関する 2 つの限度を考慮に入れる必要があります。それは、Java ヒープ・サイズとアドレス・スペース・サイズです。

Java ヒープ・サイズの限度

Java は、Java アプリケーションのコーディング作業を軽減する多数のサービスを提供しています。そのサービスの 1 つがストレージ管理です。

Java のストレージ管理では、大きなストレージ・ブロックが割り振られ、アプリケーションからの保管要求に使用されます。Java で管理されるこのストレージは、Java ヒープと呼ばれます。定期的なガーベッジ・コレクション (デフラグ) が、ヒープ内の使用されていないスペースをレクラメーション処理して、そのサイズを小さくします。CPU サイクルを節約するため、ガーベッジ・コレクションは、占有しているストレージが実際に必要になるまで待機することがよくあり、すでに使用されていないストレージを、必要以上に長い時間割り振られたままにしている (ページアウトになっている) ことに注意してください。

Java ヒープ・サイズの最大値は、rsed.envvars 内の Xmx ディレクティブで定義されます。このディレクティブが指定されていない場合、Java はデフォルト・サイズの 512 MB を使用します。256 MB 以上の値を指定する必要があります。64 ビット・モードで実行中の場合、Java は、境界より下のスペースを解放して、2 GB 境界より上のヒープを割り振ろうとします。

各 RSE スレッド・プール (クライアントのアクションをサービスします) は独立した Java アプリケーションであり、そのために専用の Java ヒープを使用します。スレッド・プールはいずれも同じ rsed.envvars 構成ファイルを使用するため、Java ヒープ・サイズの限度が同じであることに注意してください。

スレッド・プールによる Java ヒープの使用量は、接続されているクライアントが実行するアクションによって大きく異なります。最適なヒープ・サイズの限度を設定するには、ヒープの使用量を定期的にモニターすることが必要です。RSE スレッド・プールによる Java ヒープの使用量をモニターするには、**modify display process** オペレーター・コマンドを使用します。

アドレス・スペース・サイズの限度

Java アプリケーションを含むすべての z/OS アプリケーションは、アドレス・スペース内でアクティブとなるため、アドレス・スペース・サイズの限度によって制約を受けます。

必要なアドレス・スペース・サイズは、始動時に JCL の REGION パラメーターなどで指定します。ただし、システム設定によって実際のアドレス・スペース・サイズが制限されることがあります。これらの限度については、201 ページの『アドレス・スペース・サイズ』を参照してください。

- SYS1.PARMLIB(BPXPRMxx) 内の MAXASSIZE
- 開始タスクに割り当てられているユーザー ID の OMVS セグメント内の ASSIZEMAX
- システム出口 IEFUSI および IEALIMIT
- 64 ビットのアドレッシング・モードの SYS1.PARMLIB(SMFPRMxx) にある MEMLIMIT

RSE スレッド・プールは、RSE デーモンからアドレス・スペース・サイズの限度を継承します。アドレス・スペース・サイズは、Java ヒープ、Java 自体、共通ストレージ域、およびシステムがスレッド・プール・アクティビティーをサポートするために作成する全制御ブロック (スレッドごとの TCB (タスク制御ブロック) など) を収容できるだけの大きさであることが必要です。このストレージの使用量の一部は 16 MB 境界より下にあるので注意してください。64 ビット・モードで実行中の場合、Java は、境界より下のスペースを解放して、2 GB 境界より上のヒープを割り振ろうとします。

アドレス・スペース・サイズに影響する設定 (Java ヒープのサイズや 1つのスレッド・プールでサポートされるユーザー数など) を変更する前に、実際のアドレス・スペース・サイズをモニターする必要があります。Developer for System z による実際のストレージ使用量を追跡するには、通常使用しているシステム・モニター・ソフトウェアを使用します。専用のモニター・ツールがない場合は、SDSF DA ビューヤや TASID などのツール (ISPF の「Support and downloads」Web ページで入手できる無保証のシステム情報ツール) で基本情報を収集できます。

サイズ見積もりのガイドライン

すでに説明したように、Developer for System z の実際のストレージ使用量は、ユーザーのアクティビティーに大きく左右されます。一部のアクションは使用するストレージの量が一定ですが (ログオンなど)、それ以外は変動します (例えば、指定された高位修飾子を持つデータ・セットをリストする処理など)。

- RSE には、Java ヒープとすべてのシステム制御ブロックを収容できるように、2 GB のアドレス・スペースを使用してください。

- 64 ビット・モードで実行中の場合、2 GB 境界より上のストレージを RSE で実際に使用するようにしてください。
- アドレス・スペース・サイズを設定できる場所の詳細については、201 ページの『アドレス・スペース・サイズ』を参照してください。
- サンプルの `rsed.envvars` セットアップでは、スレッド・プール当たりのユーザー数が 30 に設定されています。
 - `maximum.clients=30`
 - `maximum.threads=520` ($10+17*30 = 520$ 、520 は 30 のクライアントを許可)
- サンプルの `rsed.envvars` セットアップでは、Java ヒープを 512 MB まで拡大できるようになっています。これにより、30 のクライアントそれぞれが平均で 17 MB を使用できます ($30*17 = 510$)。

RSE の始動時に、コンソール・メッセージ FEK004I で、現行の Java ヒープおよびアドレス・スペース・サイズの限度が表示されるので注意してください。

モニター中に現行の Java ヒープ・サイズが実際のワークロードに対して不十分であることがわかった場合は、以下のいずれかのシナリオを使用します。

- `rsed.envvars` の `Xmx` ディレクティブで、Java ヒープ・サイズの最大値を増やします。これを行う前に、アドレス・スペースにサイズを増やす余地があることを確認してください。
- `rsed.envvars` の `maximum.clients` ディレクティブで、スレッド・プール当たりのクライアントの最大数を減らします。RSE がサポートするクライアントの数は変わりませんが、クライアントが分散されるスレッド・プールの数が増えます。

参照用に、表 30 には、実際の Developer for System z カスタマーが、キー `rsed.envvars` 設定として使用する値を示します。これはストレージ使用量に影響を与えます。

表 30. ストレージ使用量に関する参照設定

mx (最大 Java ヒープ)	maximum.clients	主な開発のタイプ
512M	30	PL/I
512M	10	COBOL
384M	12	COBOL
800M (64 ビット)	20	指定なし

ストレージ使用量分析のサンプル

以下の各図には、次の変更を加えたデフォルトの Developer for System z セットアップに関するサンプルのリソース使用量の数値が示されています。

- `single.logon` を無効にして、RSE が 2 つ以上のスレッド・プール・アドレス・スペースを作成するのを防ぎます。
- Java ヒープ・サイズの最大値が 10 MB に設定されています。これは、最大値を小さくすることで使用率が高くなり、ヒープ・サイズの限度により早く到達するためです。

Max Heap Size=10MB and private AS Size=1,959MB

startup

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(7%) Clients(0)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2740	72
RSED	4.47	32.8M	15910
RSED8	1.15	27.4M	12612

logon 1

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2864	81
RSED	4.55	32.8M	15980
RSED8	3.72	55.9M	24128

logon 2

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(23%) Clients(2)

Jobname	Cpu time	Storage	EXCP
JMON	0.02	2944	86
RSED	4.58	32.9M	16027
RSED8	4.20	57.8M	25205

logon 3

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(37%) Clients(3)

Jobname	Cpu time	Storage	EXCP
JMON	0.02	3020	91
RSED	4.60	32.9M	16076
RSED8	4.51	59.6M	26327

logon 4

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(41%) Clients(4)

Jobname	Cpu time	Storage	EXCP
JMON	0.02	3108	96
RSED	4.61	32.9M	16125
RSED8	4.77	62.3M	27404

図 21. ログオン数 5 の場合のリソース使用量

logon 5

```
BPXM023I (STCRSE)
ProcessId(268      ) Memory Usage(41%) Clients(4)
ProcessId(33554706) Memory Usage(13%) Clients(1)
```

Jobname	Cpu time	Storage	EXCP
JMON	0.03	3184	101
RSED	4.64	32.9M	16229
RSED8	4.78	62.4M	27413
RSED9	4.60	56.6M	24065

図 22. ログオン数 5 の場合のリソース使用量 (続き)

102 ページの図 21 と 図 22 は、Java ヒープが 10 MB に設定された RSE デーモンに 5 つのクライアントがログオンしているシナリオを示しています。

- スレッド・プール (RSED8) は、開始された時点で休止状態にあり、約 27 MB を使用しています。そのうち 0.7 MB は Java ヒープ内にあります (10 MB の 7%)。
- このスレッド・プールは、最初のクライアントが接続するとアクティブになって新たに 27 MB を使用し、クライアントが接続するたびに追加で 2 MB ずつ使用します。
- ヒープ使用量が増えていることからわかるように、この接続当たりの 2 MB の一部は Java ヒープ内にあります。
- ただし、ヒープ使用量は、必要なストレージを見積もって必要以上に割り振る Java のメカニズムに依存するため、実際のパターンは存在しません。断続的なガーベッジ・コレクションによってストレージが解放されるため、傾向を見つけ出すことはさらに難しくなります。
- 内部のメカニズムによって、アクティブなスレッドに十分なヒープ・サイズを確保できるようにスレッド・プール当たりの接続数が制限されているため、5 番目の接続は新しいスレッド・プール (RSED9) に作成されます。正しく構成されたセットアップを使用している場合は、他の限度に先に到達するため (最も可能性が高いのは `rsed.envvars` 内の `maximum.clients`)、通常はこのような内部の安全策が実行されることはありません。

Max Heap Size=10MB and private AS Size=1,959MB

startup

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(7%) Clients(0)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2736	71
RSED	4.35	32.9M	15117
RSED8	1.43	27.4M	12609

logon

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2864	80
RSED	4.48	33.0M	15187
RSED8	3.53	53.9M	24125

expand large MVS tree (195 data sets)

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2864	80
RSED	4.58	33.1M	16094
RSED8	4.28	56.1M	24740

expand small PDS (21 members)

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
IBMUSER2	0.22	2644	870
JMON	0.01	2864	80
RSED	4.61	33.1M	16108
RSED8	4.40	56.2M	24937

open medium sized member (86 lines)

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
IBMUSER2	0.22	2644	870
JMON	0.01	2864	80
RSED	4.61	33.1M	16108
RSED8	8.12	62.7M	27044

図 23. PDS メンバー編集時のリソース使用量

図 23 は、Java ヒープが 10 MB の RSE デーモンに 1 つのクライアントがログオンし、PDS メンバーを編集するシナリオを示しています。

- 195 のデータ・セット名を検出したカタログ検索で約 2 MB のストレージが使用されています。Java ヒープ使用量は増えていないことから、これはすべてシステム・アクティビティによるものです。

- メンバー数 21 の PDS をオープンする操作ではスレッド・プール内のメモリーはほとんど使用されませんが、表示内容から TSO コマンド・サービスが呼び出されたことがわかります。新しいアドレス・スペース (IBMUSER2) がアクティブになり、TSO でこのユーザーに割り当てられている領域サイズを使用します。このアドレス・スペースは指定された時間内はアクティブであるため、TSO コマンド・サービスによるその後の要求に再利用できます。
- メンバーをオープンすると、高位修飾子を拡張するときと同じような数値が示されます。Java ヒープ使用量は変わりませんが、システム・アクティビティーのためにストレージが 6.5 MB 増えています。

z/OS UNIX ファイル・システム・スペースの使用量

DD ステートメントに書き込まれない Developer for System z 関連データのほとんどは、最終的に z/OS UNIX ファイルに格納されます。システム・プログラマーは、どのデータが書き込まれ、どこに格納されるかを制御できます。ただし、書き込まれるデータの量を制御することはできません。

データは、次のカテゴリーに分類することができます。

- 問題分析 (ログ・ファイルとシステム・ダンプ・ファイル)。これについては、183 ページの『第 12 章 トラブルシューティング、構成問題の』で詳しく説明しています。
- 監査。これについては 27 ページの『監査ロギング』で説明しています。
- クライアントへのプッシュ・メタデータ。これについては 133 ページの『クライアントへのプッシュ・メタデータ』で説明しています。
- 一時データ

183 ページの『第 12 章 トラブルシューティング、構成問題の』で説明したように、Developer for System z は、RSE 関連のホスト・ログを以下の z/OS UNIX ディレクトリーに書き込みます。

- /var/rdz/logs (RSE 開始タスク・ログ)
- /var/rdz/logs/\$LOGNAME (ユーザー・ログ)

デフォルトでは、エラー・メッセージと警告メッセージだけがログに書き込まれます。そのため、すべてが計画したとおりに進めば、上記のディレクトリーにはほとんど、またはまったくファイルが存在しないはずです (監査ログは対象外です)。

情報メッセージのロギングを有効にすることができますが (本来は IBM サポートの指示の下で行います)、ログ・ファイルのサイズが著しく増大します。

```

startup

$ ls -l /var/rdz/logs
total 144
-rw-rw-rw- 1 STCRSE STCGRP 33642 Jul 10 12:10 rsedaemon.log
-rw-rw-rw- 1 STCRSE STCGRP 1442 Jul 10 12:10 rseserver.log

logon

$ ls -l /var/rdz/logs
total 144
drwxrwxrwx 3 IBMUSER SYS1 8192 Jul 10 12:11 IBMUSER
-rw-rw-rw- 1 STCRSE STCGRP 36655 Jul 10 12:11 rsedaemon.log
-rw-rw-rw- 1 STCRSE STCGRP 1893 Jul 10 12:11 rseserver.log
$ ls -l /var/rdz/logs/IBMUSER
total 160
-rw-rw-rw- 1 IBMUSER SYS1 3459 Jul 10 12:11 ffs.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 ffsget.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 ffsput.log
-rw-rw-rw- 1 IBMUSER SYS1 303 Jul 10 12:11 lock.log
-rw-rw-rw- 1 IBMUSER SYS1 126 Jul 10 12:11 rmt_classloader_cache.jar
-rw-rw-rw- 1 IBMUSER SYS1 7266 Jul 10 12:11 rsecomm.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 stderr.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 stdout.log

logoff

$ ls -l /var/rdz/logs
total 80
drwxrwxrwx 3 IBMUSER SYS1 8192 Jul 10 12:11 IBMUSER
-rw-rw-rw- 1 STCRSE STCGRP 36655 Jul 10 12:11 rsedaemon.log
-rw-rw-rw- 1 STCRSE STCGRP 2208 Jul 10 12:11 rseserver.log
$ ls -l /var/rdz/logs/IBMUSER
total 296
-rw-rw-rw- 1 IBMUSER SYS1 6393 Jul 10 12:11 ffs.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 ffsget.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 ffsput.log
-rw-rw-rw- 1 IBMUSER SYS1 609 Jul 10 12:11 lock.log
-rw-rw-rw- 1 IBMUSER SYS1 126 Jul 10 12:11 rmt_classloader_cache.jar
-rw-rw-rw- 1 IBMUSER SYS1 45157 Jul 10 12:11 rsecomm.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 stderr.log
-rw-rw-rw- 1 IBMUSER SYS1 176 Jul 10 12:11 stdout.log

stop

$ ls -l /var/rdz/logs
total 80
drwxrwxrwx 3 IBMUSER SYS1 8192 Jul 10 12:11 IBMUSER
-rw-rw-rw- 1 STCRSE STCGRP 36655 Jul 10 12:11 rsedaemon.log
-rw-rw-rw- 1 STCRSE STCGRP 2490 Jul 10 12:12 rseserver.log

```

図 24. z/OS UNIX ファイル・システム・スペースの使用量

図 24 は、デバッグ・レベル 2 (情報メッセージ) を使用する場合は z/OS UNIX ファイル・システム・スペースの最小使用量を示しています。

- 開始タスク・ログは開始後に 34 KB を使用し、ユーザーのログオン、ログオフ、またはオペレーター・コマンドの発行に伴って少しずつ増大します。
- クライアント・ログ・ディレクトリーはログオン後に 11 KB を使用し、ユーザーが作業を開始すると一定のペースで増大します (サンプルには示されていません)。
- ログオフによって新たに 40 KB がユーザー・ログに追加されて、51 KB になります。

監査ログ以外のログ・ファイルは、再始動 (RSE 開始タスクの場合) またはログオン (クライアントの場合) のたびに上書きされて、合計サイズを抑えます。この動作は、rsed.envvars 内の keep.last.log ディレクティブによって多少変わります。このディレクティブでは、前のログのコピーを残すように RSE に指示できるからです。それより古いコピーは必ず除去されます。

監査がアクティブであるときに、監査ログ・ファイルを保持しているファイル・システムのフリー・スペースが不足してくると、警告メッセージがコンソールに送信されます。このコンソール・メッセージ (FEK103E) は、スペース不足の問題が解決されるまで定期的に繰り返されます。RSE が生成するコンソール・メッセージのリストについては、「[ホスト構成ガイド](#)」 (SC88-5663)の『コンソール・メッセージ』を参照してください。

表 31 に示す定義は、ログ・ディレクトリーに書き込まれるデータおよびディレクトリーのロケーションを制御します。

表 31. ログ出力ディレクティブ

ロケーション	ディレクティブ	機能
resecomm.properties	debug_level	デフォルトのログ詳細レベルを設定します。
rsed.envvars	keep.last.log	開始/ログオン前に、前のログのコピーを保存します。
rsed.envvars	enable.audit.log	クライアント・アクションの監査トレースを保存します。
rsed.envvars	enable.standard.log	スレッド・プール (1 つまたは複数) の stdout および stderr ストリームをログ・ファイルに書き込みます。
rsed.envvars	DSTORE_TRACING_ON	DataStore アクションのロギングを有効にします。
rsed.envvars	DSTORE_MEMLOGGING_ON	DataStore メモリ使用量のロギングを有効にします。
オペレーター・コマンド	modify rsecommlog <level>	rsecomm.log のログ詳細レベルを動的に変更します。
オペレーター・コマンド	modify rsedaemonlog <level>	rsedaemon.log のログ詳細レベルを動的に変更します。
オペレーター・コマンド	modify rseserverlog <level>	rseserver.log のログ詳細レベルを動的に変更します。
オペレーター・コマンド	modify rsestandardlog {onloff}	std*.*.log の更新を動的に変更します。
rsed.envvars	daemon.log	RSE 開始タスク・ログおよび監査ログのホーム・パス。
rsed.envvars	user.log	ユーザー・ログのホーム・パス
rsed.envvars	CGI_ISPWORK	ISPF クライアント・ゲートウェイ・ログのホーム・パス
rsed.envvars	TMPDIR	IVP ログのディレクトリー
rsed.envvars	_CEE_DMPTARG	Java ダンプのディレクトリー

これに加えて、Developer for System z は、ISPF クライアント・ゲートウェイなどの必要なソフトウェアとともに、一時データを /tmp および /var/rdz/WORKAREA に書き込みます。ユーザー・アクションの結果としてここに書き込まれるデータの量は予測不能であるため、これらのディレクトリーを保持するファイル・システムに十分なフリー・スペースを確保しておく必要があります。

Developer for System z は、これらの一時ファイルのクリーンアップを常時試みますが、「ホスト構成ガイド」(SC88-5663)の『(オプション) WORKAREA と /tmp クリーンアップ』で説明しているように、手動でのクリーンアップも随時実行できます。

表 32 の定義は、一時データ・ディレクトリーをどこに置くかを制御します。

表 32. 一時出力ディレクティブ

ロケーション	ディレクティブ	機能
rsed.envvars	CGI_ISPWORK	一時データのホーム・パス
rsed.envvars	TMPDIR	一時データのディレクトリー

主要なリソース定義

/etc/rdz/rsed.envvars

rsed.envvars で定義されている環境変数は、RSE、Java、および z/OS UNIX によって使用されます。Developer for System z 付属のサンプル・ファイルは、Developer for System z のオプション・コンポーネントを必要としない小規模から中規模のインストール済み環境を対象としています。サンプル・ファイルに定義されている各変数については、「ホスト構成ガイド」(SC88-5663)の

『rsed.envvars、RSE 構成ファイル』"で説明していますが、以下の変数には特に注意が必要です。

_RSE_JAVA_OPTS="\$_RSE_JAVA_OPTS -Xms128m -Xmx512m"

初期 (Xms) および最大 (Xmx) ヒープ・サイズを設定します。デフォルトはそれぞれ、128M と 512M です。希望するヒープ・サイズ値を強制的に使用させるには、変更してください。このディレクティブがコメント化されている場合は、Java のデフォルト値が使用されます。デフォルト値はそれぞれ 4M と 512M です。

#_RSE_JAVA_OPTS="\$_RSE_JAVA_OPTS -Dmaximum.clients=30"

1 つのスレッド・プールでサービスできるクライアントの最大数。デフォルトは 30 です。1 スレッド・プール当たりのクライアント数を制限するには、コメント解除してカスタマイズします。他の制限のために、RSE がこの限度に到達しない場合もあることに注意してください。

#_RSE_JAVA_OPTS="\$_RSE_JAVA_OPTS -Dmaximum.threads=520"

新規クライアントを許可するための、1 つのスレッド・プールでのアクティブ・スレッド数の最大値。デフォルトは、520 です。1 スレッド・プール当たりのクライアント数を、使用中のスレッド数に基づいて制限するには、コメント解除し

てカスタマイズします。それぞれのクライアント接続が複数 (17 以上) のスレッドを使用すること、および他の制限のために RSE がこの限度に到達しない場合があることに注意してください。

注: この値は、SYS1.PARMLIB(BPXPRMxx) での MAXTHREADS および MAXTHREADTASKS の設定より小さくする必要があります。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dminimum.threadpool.process=1"

アクティブ・スレッド・プールの最小数。デフォルトは 1 です。少なくともリストされた数のスレッド・プール・プロセスを開始するには、コメント解除してカスタマイズします。スレッド・プール・プロセスは、RSE サーバー・スレッドのロード・バランシングに使用されます。より多くの新規プロセスが必要になった場合は、その時点で新規プロセスが開始されます。前もって新規プロセスを開始しておく、接続遅延を回避できますが、アイドル時間中に使用されるリソースが増えます。

注: single.logon ディレクティブがアクティブな場合、minimum.threadpool.process が 1 に設定されているとしても、少なくとも 2 つのスレッド・プールが開始されます。rsed.envvars 内の single.logon のデフォルトの設定はアクティブです。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dmaximum.threadpool.process=100"

アクティブ・スレッド・プールの最大数。デフォルトは 100 です。スレッド・プール・プロセスの数を制限するには、コメント解除してカスタマイズします。スレッド・プール・プロセスは、RSE サーバー・スレッドのロード・バランシングに使用されます。このため、スレッドを制限すると、アクティブなクライアント接続の数も制限されます。

SYS1.PARMLIB(BPXPRMxx)

RSE は Java アプリケーションであるため、z/OS UNIX 環境でアクティブになります。このため、z/OS UNIX の環境とファイル・システムを制御するパラメーターを含んでいる BPXPRMxx は、非常に重要な parmlib メンバーとなります。BPXPRMxx については、「MVS 初期設定およびチューニング解説書」(SA88-8564) で説明されています。Developer for System z に影響することがわかっているディレクティブは以下のとおりです。

MAXPROCSYS(nnnnn)

システムで許可されるプロセスの最大数を指定します。

値の範囲: nnnnn は 5 から 32767 までの 10 進値です。
デフォルト: 900

MAXPROCUSER(nnnnn)

プロセスが作成された方法に関係なく、単一の z/OS UNIX ユーザー ID が同時にアクティブにしておくことができるプロセスの最大数を指定します。

値の範囲: nnnnn は 3 から 32767 までの 10 進値です。
デフォルト: 25

注:

- RSE プロセスはいずれも同じ z/OS UNIX ユーザー ID (RSE デーモンに割り当てられているユーザーの ID) を使用します。これは、すべてのクライアントが RSE プロセス内のスレッドとして実行されるためです。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の PROCUSERMAX 変数で設定することもできます。

MAXTHREADS(nnnnnn)

単一のプロセスが同時にアクティブにしておくことができる pthread_created スレッドの最大数を指定します。これには、実行中のスレッド、キューに入れられたスレッド、および終了してまだ切り離されていないスレッドが含まれます。値 0 を指定すると、アプリケーションで pthread_create が使用されなくなります。

値の範囲: nnnnnn は 0 から 100000 までの 10 進値です。

デフォルト: 200

注:

- 各クライアントは、RSE スレッド・プール・プロセス内で少なくとも 17 のスレッドを使用し、プロセス内では複数のクライアントがアクティブになります。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の THREADSMAX 変数で設定することもできます。設定された THREADSMAX 値は、MAXTHREADS と MAXTHREADTASKS の両方に使用されます。

MAXTHREADTASKS(nnnnn)

単一プロセスが pthread_created スレッドに対して同時にアクティブにしておくことができる MVS タスクの最大数を指定します。

値の範囲: nnnnn は 0 から 32768 までの 10 進値です。

デフォルト: 1000

注:

- アクティブなスレッドそれぞれに MVS タスク (TCB、タスク制御ブロック) があります。
- 同時 MVS タスクそれぞれに追加のストレージが必要で、その一部は 16 MB 境界より下に位置している必要があります。
- 各クライアントは、RSE スレッド・プール・プロセス内で少なくとも 17 のスレッドを使用し、プロセス内では複数のクライアントがアクティブになります。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の THREADSMAX 変数で設定することもできます。設定された THREADSMAX 値は、MAXTHREADS と MAXTHREADTASKS の両方に使用されます。

MAXUIDS(nnnnn)

同時に操作を実行できる z/OS UNIX ユーザー ID (UID) の最大数を指定します。

値の範囲: nnnnn は 1 から 32767 までの 10 進値です。
デフォルト: 200

MAXASSIZE(nnnnn)

新しいプロセスの初期値として設定される RLIMIT_AS リソース値を指定します。RLIMIT_AS は、アドレス・スペースの領域サイズを示します。

値の範囲: nnnnn は 10485760 (10 MB) から 2147483647 (2 GB) までの 10 進値です。
デフォルト: 209715200 (200 MB)

注:

- この値は 2G に設定する必要があります。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の ASSIZEMAX 変数で設定することもできます。

MAXFILEPROC(nnnnnn)

単一プロセスが同時にアクティブにするか、割り振ることができる、ファイル、ソケット、ディレクトリー、およびその他のファイル・システム・オブジェクトの記述子の最大数を指定します。

値の範囲: nnnnnn は 3 から 524287 までの 10 進値です。
デフォルト: 64000

注:

- スレッド・プールでは、そのすべてのクライアント・スレッドが単一のプロセスに入っています。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の FILEPROCMAX 変数で設定することもできます。

MAXMMAPAREA(nnnnn)

z/OS UNIX ファイルのメモリー・マッピングに割り振ることのできるデータ・スペース・ストレージ・スペースの最大量 (ページ単位) を指定します。メモリー・マッピングがアクティブになるまで、ストレージは割り振られません。

値の範囲: nnnnn は 1 から 16777216 までの 10 進値です。
デフォルト: 40960

注: この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の MMAPAREAMAX 変数で設定することもできます。

元の BPXPRMxx 変数の値を動的に (次回の IPL まで) 増減させるには、オペレーター・コマンド **SETOMVS** または **SET OMVS** を使用します。永続的な変更を加える場合は、IPL に使用される BPXPRMxx メンバーを編集します。これらのオペレーター・コマンドの詳細については、「MVS システム・コマンド」(SA88-8593) を参照してください。

以下の定義は、NETWORK ステートメントのサブパラメーターです。

MAXSOCKETS(nnnnnnnn)

このアドレス・ファミリーに対してこのファイル・システムでサポートされる最大ソケット数を指定します。これはオプション・パラメーターです。

値の範囲: nnnnnnnn は 0 から 16777215 までの 10 進値です。

デフォルト: 100

INADDRANYCOUNT(nnnn)

システムが PORT 0、INADDR_ANY バインドに使用するために予約する、INADDRANYPORT パラメーターで指定したポート番号から始まるポートの数を指定します。この値は、CINET (複数の TCP/IP スタック) にのみ必要です。

値の範囲: nnnn は 1 から 4000 までの 10 進値です。

デフォルト: INADDRANYPORT と INADDRANYCOUNT がどちらも

指定されていない場合、INADDRANYCOUNT の

デフォルトは 1000 です。

それ以外の場合はポートが予約されません (0)。

さまざまなリソース定義

サーバー JCL での EXEC カード

以下の定義は、Developer for System z サーバーの JCL の EXEC カードに追加することをお勧めします。

REGION=0M

RSE デーモンおよび JES ジョブ・モニターの開始タスク (それぞれ RSED と JMON) には REGION=0M を指定することをお勧めします。そうすることで、アドレス・スペース・サイズが、使用可能な専用ストレージ、あるいは IEFUSI または IEALIMIT システム出口によってのみ制限されます。IBM では、z/OS UNIX アドレス・スペースには RSE デーモンのようにこれらの出口を使用しないよう強く推奨しています。

TIME=NOLIMIT

すべての Developer for System z サーバーに TIME=NOLIMIT を使用することをお勧めします。これは、すべての Developer for System z クライアントの CPU 時間がサーバー・アドレス・スペース内で集計されるためです。

FEK.#CUST.PARMLIB(FEJJCNFG)

FEJJCNFG で定義されている環境変数は、JES ジョブ・モニターで使用されます。Developer for System z 付属のこのサンプル・ファイルは、小規模から中規模のインストール済み環境を対象としています。サンプル・ファイルに定義されている各変数については、「ホスト構成ガイド」(SC88-5663) の「『FEJJCNFG、JES ジョブ・モニター構成ファイル』」で説明していますが、以下の変数には特に注意が必要です。

MAX_THREADS

1 つの JES ジョブ・モニターを一度に使用できるユーザーの最大数。デフォルトは 200 です。最大値は 2147483647 です。この数を大きくすると、JES ジョブ・モニターのアドレス・スペースのサイズを大きくしなければならない場合があります。

SYS1.PARMLIB(IEASYSxx)

IEASYSxx はシステム・パラメーターを保持するもので、「MVS 初期設定およびチューニング解説書」(SA88-8564) に説明があります。Developer for System z に影響することがわかっているディレクティブは以下のとおりです。

MAXUSER=nnnnn

このパラメーターは、ほとんどの条件下で、特定の IPL 時に同時に実行できるジョブおよび開始タスクの数を制限するためにシステムが使用する値を指定します。

値の範囲: nnnnn は 0 から 32767 までの 10 進値です。

MAXUSER、RSVSTRT、および RSVNONR の各システム・パラメーターに指定された値の合計は 32767 以下でなければならないので注意してください。

デフォルト値: 255

SYS1.PARMLIB(IVTPRMxx)

IVTPRMxx は通信ストレージ・マネージャー (CSM) のパラメーターを設定するもので、「MVS 初期設定およびチューニング解説書」(SA88-8564) に説明があります。Developer for System z に影響することがわかっているディレクティブは以下のとおりです。

FIXED MAX(maxfix)

固定 CSM バッファ専用ストレージの最大量を定義します。

値の範囲: 1024K から 2048M までの値。

デフォルト: 100M

ECSA MAX(maxecsa)

ECSA CSM バッファ専用ストレージの最大量を定義します。

値の範囲: 1024K から 2048M までの値。

デフォルト: 100M

SYS1.PARMLIB(ASCHPMxx)

ASCHPMxx parmlib メンバーは、ASCH トランザクション・スケジューラーのスケジューリング情報を格納するもので、「MVS 初期設定およびチューニング解説書」(SA88-8564) に説明があります。Developer for System z に影響することがわかっているディレクティブは以下のとおりです。

MAX(nnnnn)

CLASSADD 定義のオプション・パラメーター。トランザクション・イニシエーターの特定のクラスに対して許可される APPC トランザクション・イニシエーターの最大数を指定します。この限度に到達すると、新しいアドレス・スペースが作成されなくなり、着信要求がキューに入れられて、既存のイニシエーター・アドレス・スペースが使用可能になるまで待機します。この値は、ご使用のシステムで許可されている最大アドレス・スペース数を超えないようにする必要があります。また、同じくアドレス・スペースを必要とするシステム上の競合製品に留意してください。

値の範囲: nnnnn は 1 から 64000 までの 10 進値です。
デフォルト: 1

注: APPC を使用して TSO コマンド・サービスを開始する場合は、使用される
トランザクション・クラスに、Developer for System z の各同時ユーザーに 1
つずつイニシエーターを許可できるだけの十分な数のトランザクション・イニシ
エーターがあることが必要です。

モニター

ユーザーのワークロードによってシステム・リソースの必要性が変わる可能性があ
るため、ユーザーの要件に応じて Rational Developer for System z とシステムの構
成を調整できるように、定期的にシステムをモニターしてリソース使用量を測定す
る必要があります。このモニター処理に役立つコマンドを以下に示します。

RSE のモニター

RSE スレッド・プールは、Developer for System z におけるユーザー・アクティビ
ティのフォーカル・ポイントであるため、最適に利用されているかモニターする
必要があります。通常のシステム・モニター・ツールでは収集できない情報は、
RSE デーモンに照会することができます。

- アドレス・スペース固有のデータ (使用されている実ストレージや CPU 時間な
ど) を収集するには、RMF™ などの通常のシステム・モニター・ツールを使用し
ます。専用のモニター・ツールがない場合は、SDSF DA ビューや TASID など
のツール (ISPF の「Support and downloads」Web ページで入手できる無保証のシ
ステム情報ツール) で基本情報を収集できます。
- RSE デーモンは、使用可能なアドレス・スペース・サイズと Java ヒープ・サイ
ズを、始動時にコンソール・メッセージ FEK004I で通知します。

```
FEK004I RseDaemon: Max Heap Size=65MB and private AS Size=1,959MB
```

- **MODIFY RSED,APPL=DISPLAY PROCESS** オペレーター・コマンドは、RSE
スレッド・プール・プロセスを表示します。「Memory Usage」フィールドには、
定義された Java ヒープが実際にどの程度使用されているかが示されます。この
コマンドについて詳しくは、「ホスト構成ガイド」(SC88-5663) の『オペレータ
ー・コマンド』を参照してください。

```
f rsed,appl=d p  
BPXM023I (STCRSE)  
ProcessId(16777456) Memory Usage(33%) Clients(4) Order(1)
```

DISPLAY PROCESS 変更コマンドの DETAIL オプションを使用すると、詳細情
報が表示されます。

```
f rsed,appl=d p,detail  
BPXM023I (STCRSE)  
ProcessId(33555087) ASId(002E) JobName(RSED8) Order(1)  
PROCESS LIMITS:  CURRENT  HIGHWATER    LIMIT  
JAVA HEAP USAGE(%)  10         56         100  
CLIENTS              0          25         30  
MAXFILEPROC          83        103        64000  
MAXPROCUSER          97         99         200  
MAXTHREADS           9         14        1500  
MAXTHREADTASKS       9         14        1500
```

「**DISPLAY PROCESS**」 変更コマンドの CPU オプションにより、スレッド・プール内のスレッドごとに累積 CPU 使用時間がミリ秒で表示されます。

```
f rsed,appl=d p,cpu
BPXM023I (STCRSE)
ProcessId(33555087) ASId(002E) JobName(RSED8) Order(1)
USERID  THREAD-ID      TCB@      ACC_TIME TAG
STCRSE   0EDE54000000000 005E6B60      822 1/ThreadPoolProcess
STCRSE   0EDE87000000000 005E69C8        001
STCRSE   0EDE98000000000 005E6518      1814
STCRSE   0EDEBA000000000 005E66B0      2305
STCRSE   0EDECB000000000 005E62F8        001
STCRSE   0EDED0000000000 005E60D8        001
STCRSE   0EDF86000000000 005C2BF8      628 6/ThreadPoolMonitor$Memory
UsageMonitor
STCRSE   0EDF97000000000 005C2D90        003 7/ThreadPoolMonitor
IBMUSER  0EE2C700000000024 005C08B0        050 38/JESMiner
IBMUSER  0EE2B600000000026 005C0690        004 40/FAMiner
IBMUSER  0EE30B00000000027 005C0250        002 41/LuceneMiner
IBMUSER  0EE31C00000000028 005C0030        002 42/CDTParserMiner
IBMUSER  0EE32D00000000029 005BDE00        002 43/MVSLuceneMiner
IBMUSER  0EE33E0000000002A 005BDBE0        002 44/CDTMVSParserMiner
```

- RSE スレッド・プール・プロセスの終了時には、その RSE スレッド・プール・プロセスに対して 「**DISPLAY PROCESS,DETAIL**」 変更コマンドを発行したときと同様に、リソース使用の詳細な統計が表示されます。最高水準点は、RSE スレッド・プール・プロセスの存在期間中に同時に使用されるリソースの最大量を示します。これにより、システム・チューナーは RSE に割り当てられるリソースが過大に割り振りされているか、過小に割り振りされているかを判断できます。

z/OS UNIX のモニター

Developer for System z に関する z/OS UNIX の限度のほとんどは、オペレーター・コマンドを使用して表示できます。一部のコマンドでは、特定の限度に関する現在の使用率と最高水準点も示されます。これらのコマンドの詳細については、「MVS システム・コマンド」(SA88-8593) を参照してください。

- SYS1.PARMLIB(BPXPRMxx) 内の LIMMSG(ALL) ディレクティブは、parmlib のいずれかの限度に到達しそうなときにコンソール・メッセージ (BPXI040I) を表示するよう z/OS UNIX に指示します。LIMMSG のデフォルト値は NONE です。この値が指定されると機能は無効になります。この機能を動的にアクティブにする (次の IPL まで) には、オペレーター・コマンド **SETOMVS LIMMSG=ALL** を使用します。このディレクティブの詳細については、「MVS 初期設定およびチューニング解説書」(SA88-8564) を参照してください。
- **DISPLAY OMVS,OPTIONS** オペレーター・コマンドは、動的に設定できる z/OS UNIX ディレクティブの現行値を表示します。

```
d omvs,o
BPX0043I 13.10.16 DISPLAY OMVS 066
OMVS      000D ETC/INIT WAIT  OMVS=(M7)
CURRENT UNIX CONFIGURATION SETTINGS:
MAXPROCSYS      =      256    MAXPROCUSER      =      16
MAXFILEPROC     =      256    MAXFILESIZE       =  NOLIMIT
MAXCPUPTIME     =      1000   MAXUIDS         =      200
MAXPTYS         =      256
MAXMMAPAREA     =      256    MAXASSIZE        = 209715200
MAXTHREADS      =      200    MAXTHREADTASKS   =      1000
MAXCORESIZE     =    4194304   MAXSHAREPAGES =      4096
IPCMSGQBYTES    = 2147483647  IPCMSGQNUM    =    10000
```

```

IPCMSGNIDS      =      500      IPCSEMNIDS      =      500
IPCSEMNOPS      =      25      IPCSEMNSEMS      =      1000
IPCSHMMPAGES    =      25600    IPCSHMNIDS      =      500
IPCSHMNSEGS     =      500     IPCSHMSPAGES    =      262144
SUPERUSER       = BPXROOT      FORKCOPY        = COW
STEPLIBLIST     =
USERIDALIASTABLE=
SERV_LINKLIB    = POSIX.DYN SERV.LOADLIB  BPXLK1
SERV_LPALIB     = POSIX.DYN SERV.LOADLIB  BPXLK1
PRIORITYPG VALUES: NONE
PRIORITYGOAL VALUES: NONE
MAXQUEUEDSIGs   =      1000     SHRLIBRGNSIZE   =      67108864
SHRLIBMAXPAGES  =      4096     VERSION         = /
SYSCALL COUNTS  = NO           TTYGROUP        = TTY
SYSPLEX         = NO           BRM SERVER        = N/A
LIMMSG          = NONE         AUTOCVT         = OFF
RESOLVER PROC   = DEFAULT
AUTHPGMLIST     = NONE
SWA             = BELOW

```

- **DISPLAY OMVS,LIMITS** オペレーター・コマンドは、現在設定されている z/OS UNIX システム・サービスの parmlib 限度、それらの最高水準点、および現行のシステム使用量に関する情報を表示します。

```

d omvs,l
BPX0051I 14.05.52 DISPLAY OMVS 904
OMVS      0042 ACTIVE          OMVS=(69)
SYSTEM WIDE LIMITS:          LIMMSG=SYSTEM

```

	CURRENT USAGE	HIGHWATER USAGE	SYSTEM LIMIT
MAXPROCSYS	1	4	256
MAXUIDS	0	0	200
MAXPTYs	0	0	256
MAXMMAPAREA	0	0	256
MAXSHAREPAGES	0	10	4096
IPCMSGNIDS	0	0	500
IPCSEMNIDS	0	0	500
IPCSHMNIDS	0	0	500
IPCSHMSPAGES	0	0	262144 *
IPCMSGQBYTES	---	0	262144
IPCMSGQMNUM	---	0	10000
IPCSHMMPAGES	---	0	256
SHRLIBRGNSIZE	0	0	67108864
SHRLIBMAXPAGES	0	0	4096

このコマンドで PID=processid キーワードを指定すると、個々のプロセスの最高水準点と現行の使用量が表示されます。

```

d,omvs,l,pid=16777456
BPX0051I 14.06.28 DISPLAY OMVS 645
OMVS      000E ACTIVE          OMVS=(76)
USER      JOBNAME  ASID      PID      PPID STATE   START   CT_SECS
STCRSE    RSED8    007E     16777456  67109106 HF----- 20.00.56 113.914
LATCHWAITPID=      0 CMD=java -Ddaemon.log=/var/rdz/logs -
PROCESS LIMITS:          LIMMSG=NONE

```

	CURRENT USAGE	HIGHWATER USAGE	PROCESS LIMIT
MAXFILEPROC	83	103	256
MAXFILESIZE	---	---	NOLIMIT
MAXPROCUSER	97	99	200
MAXQUEUEDSIGs	0	1	1000
MAXTHREADS	9	14	200
MAXTHREADTASKS	9	14	1000
IPCSHMNSEGS	0	0	500
MAXCORESIZE	---	---	4194304
MAXMEMLIMIT	0	0	16383P

- **DISPLAY OMVS,PFS** オペレーター・コマンドは、z/OS UNIX 構成に現在組み込まれている各物理ファイル・システム (TCP/IP スタックを含む) に関する情報を表示します。

```
d omvs,p
BPX0046I 14.35.38 DISPLAY OMVS 092
OMVS 000E ACTIVE OMVS=(33)
PFS CONFIGURATION INFORMATION
PFS TYPE DESCRIPTION ENTRY MAXSOCK OPNSOCK HIGHUSED
TCP SOCKETS AF_INET EZBPFINI 50000 244 8146
UDS SOCKETS AF_UNIX BPXTUINI 64 6 10
ZFS LOCAL FILE SYSTEM IOEFSCM
14:32.00 RECYCLING
HFS LOCAL FILE SYSTEM GFUAINIT
BPXFTCLN CLEANUP DAEMON BPXFTCLN
BPXFTSYN SYNC DAEMON BPXFTSYN
BPXFPINT PIPE BPXFPINT
BPXFCSIN CHAR SPECIAL BPXFCSIN
NFS REMOTE FILE SYSTEM GFSCINIT
PFS NAME DESCRIPTION ENTRY STATUS FLAGS
TCP41 SOCKETS EZBPFINI ACT CD
TCP42 SOCKETS EZBPFINI ACT
TCP43 SOCKETS EZBPFINI INACT SD
TCP44 SOCKETS EZBPFINI INACT
PFS PARM INFORMATION
HFS SYNCDEFAULT(60) FIXED(50) VIRTUAL(100)
CURRENT VALUES: FIXED(55) VIRTUAL(100)
NFS biod(6)
```

- **DISPLAY OMVS,PID=processid** オペレーター・コマンドは、特定のプロセスのスレッド情報を表示します。

```
d omvs,pid=16777456
BPX0040I 15.30.01 DISPLAY OMVS 637
OMVS 000E ACTIVE OMVS=(76)
USER JOBNAME ASID PID PPID STATE START CT_SECS
STCRSE RSED8 007E 16777456 67109106 HF---- 20.00.56 113.914
LATCHWAITPID= 0 CMD=java -Ddaemon.log=/var/rdz/logs -
THREAD_ID TCB@ PRI_JOB USERNAME ACC_TIME SC STATE
0E08A00000000000 005E6DF0 OMVS .927 RCV FU
0E08F00000000001 005E6C58 .001 PTX JYNV
0E09300000000002 005E6AC0 7.368 PTX JYNV
0E0CB00000000008 005C2CF0 OMVS 1.872 SEL JFNV
0E192000000003CE 005A0B70 OMVS IBMUSER 14.088 POL JFNV
0E18D000000003CF 005A1938 IBMUSER .581 SND JYNV
```

ネットワークのモニター

ホストに接続している多数のクライアントをサポートしている場合は、Developer for System z だけでなく、ネットワーク・インフラストラクチャーでもワークロードを処理することが必要です。ネットワーク管理は、Developer for System z 資料が扱う範囲外の対象で、これに関しては幅広くかつ数多くこれまでに解説されています。したがって、以下の指針のみを示しておきます。

- **DISPLAY NET,CSM** オペレーター・コマンドでは、通信ストレージ・マネージャー (CSM) で管理されているストレージの使用状況をモニターすることができます。このコマンドを使用して、ECSA およびデータ・スペース・ストレージ・プールに CSM ストレージがどの程度使用されているかを確認できます。詳細については、「*Communications Server SNA オペレーション*」(SC88-8930) を参照してください。

z/OS UNIX ファイル・システムのモニター

Developer for System z は、z/OS UNIX ファイル・システムを使用して、ログや一時ファイルなどのさまざまなタイプのデータを保管します。z/OS UNIX の **df** コマンドを使用すると、基礎となる HFS または zFS データ・セットの次のエクステントを作成する前に、まだ使用可能なファイル記述子の数とフリー・スペースの残量を確認することができます。

```
$ df
Mounted on      Filesystem      Avail/Total      Files      Status
/tmp            (OMVS.TMP)      1393432/1396800  4294967248 Available
/u/ibmuser      (OMVS.U.IBMUSER) 1248/1728        4294967281 Available
/usr/lpp/rdz    (OMVS.LPP.FEK)   3062/43200       4294967147 Available
/var            (OMVS.VAR)      27264/31680      4294967054 Available
```

サンプル・セットアップ

次のサンプル・セットアップでは、以下の要件をサポートするために必要な構成を示します。

- 同時クライアント接続数は 500
- 同時 MVS ビルド数 (バッチ・ジョブ) は 300
- 同時 CARMA 接続数 (CRASTART 始動方式を使用) は 200
- 3 時間の非アクティブ・タイムアウト
- z/OS UNIX の使用は不許可
- SCLM Developer Toolkit と File Manager Integration は使用しない
- Java ヒープの平均使用量を 20 MB と予測
- ユーザーに固有の z/OS UNIX UID を付与

スレッド・プールの数

デフォルトでは、Developer for System z は単一のスレッド・プールに 30 ユーザーを追加しようとします。ただし上記の要件では、非アクティブ・タイムアウトをアクティブにするように指定しています。このために、接続されているクライアントごとに 1 つずつスレッドが追加されることが 95 ページの表 28 からわかります。このスレッドはタイマー・スレッドであるため、常にアクティブです。したがって $10+30*(17+1)=550$ となり、かつ `maximum.threads` がデフォルトで 520 に設定されているので、RSE は単一スレッド・プールに 30 ユーザーを配置できなくなります。

`maximum.threads` を増やすことはできますが、ユーザー当たりの Java ヒープを平均 20 MB にするという要件があるため、`maximum.clients` を 25 ($10+25*18 = 460$) まで下げることにしました。これで Java ヒープ・サイズの最大値がデフォルトの 512 MB 以内に収まります ($20*25 = 500$)。

スレッド・プール当たりのクライアント数が 25 であり、サポートの必要な接続数が 500 であることから、必要なスレッド・プール・アドレス・スペースの数は 20 であることがわかりました。

最小限度の特定

この章で説明した数式と、このセクションの冒頭で示した基準を使用して、対応が必要なリソース使用量を特定することができます。

- アドレス・スペースの数 - 最大

$$3 + A + N*(x + y + z) + (2 + N*0.01)$$

$$3 + 20 + 500*1 + 200*1 + 300*1 + (2 + 500*0.01) = 1030$$

- アドレス・スペースの数 - ユーザー単位

$$x + y + z$$

$$1 + 1 + 1 = 3$$

- プロセスの数 - 最大

$$5 + 2*A + N*(x + y + z) + (10 + N*0.05)$$

$$5 + 2*20 + 500*2 + 200*1 + 300*0 + (10 + 500*0.05) = 1570$$

- プロセス数 - STCRSE

$$4 + 2*A$$

$$4 + 2*20 = 44$$

- プロセスの数 - ユーザー単位

$$(x + y + z) + 5*s$$

$$(2 + 1 + 0) + 5*0 = 3$$

- スレッドの数 - RSE スレッド・プール

$$10 + N*(17 + x + y + z) + (20 + N*0.1)$$

$$10 + 25*(17 + 1 + 4 + 0) + (20 + 25*0.1) = 583$$

- スレッドの数 - JES ジョブ・モニター

$$3 + N$$

$$3 + 500 = 503$$

- ユーザー ID の数

$$500 + 2 = 502$$

追加されている 2 つのユーザー ID は、Developer for System z 開始タスクのユーザー ID である STCJMON および STCRSE です。

限度の定義

リソース使用量の数値が判明したので、限度を指定するディレクティブを適切な値でカスタマイズできます。

- /etc/rdz/rsed.envvars
 - Xmx512m

変更なし

- Dmaximum.clients=25
- Dmaximum.threads=520

変更なし

- Dminimum.threadpool.process=10

この変更はオプションです。RSE は必要に応じて新しいスレッド・プールを開始します。

- DHIDE_ZOS_UNIX=true
- DDSTORE_IDLE_SHUTDOWN_TIMEOUT=10800000
- FEK.#CUST.PARMLIB(FEJJCNFG)
 - MAX_THREADS=503
- SYS1.PARMLIB(BPXPRMxx)
 - MAXPROCSYS(2500)

最小は 1572、Developer

for System z

以外のタスク用にバッファを追加

- MAXPROCUSER(80)

最小は 44、RSE スレッド・プールが、予測された 25 より少ないクライアントをサポートする場合、バッファを追加

- MAXTHREADS(1500)

ユーザー ID STCRSE の OMVS セグメント内の THREADSMAX を使用して RSE の限度 (最小は 582) が設定されている場合、必ず最小は 503 (JES ジョブ・モニター用)

- MAXTHREADTASKS(1500)

ユーザー ID STCRSE の OMVS セグメント内の THREADSMAX を使用して RSE の限度 (最小は 582) が設定されている場合、必ず最小は 503 (JES ジョブ・モニター用)

- MAXUIDS(700)

最小は 503、Developer

for System z

以外のタスク用にバッファを追加

- MAXASSIZE(209715200)

変更なし (システム・デフォルトは 200 MB)、ここでは

ユーザー ID STCRSE の OMVS セグメント内で ASSIZEMAX を使用

- SYS1.PARMLIB(IEASYSxx)
 - MAXUSER=2000

最小は 1030、Developer
for System z
以外のタスク用にバッファを追加

- ユーザー ID STCRSE の OMVS セグメント
 - ASSIZEMAX(2147483647)

2 GB

リソース使用量のモニター

119 ページの『限度の定義』の説明に従ってシステム限度を定義したら、Developer for System z によるリソース使用量のモニターを開始して、変数の調整が必要かどうかを確認できます。122 ページの図 25 は、499 人のユーザーがログオンした後のリソース使用量を示しています。(この図の例はログオンだけを示しています。ユーザー・アクションは示されていません)。

```

F RSED,APPL=D P
BPXM023I (STCRSE)
ProcessId(83886168) Memory Usage(17%) Clients(25) Order(1)
ProcessId(91 ) Memory Usage(17%) Clients(25) Order(2)
ProcessId(122 ) Memory Usage(17%) Clients(25) Order(3)
ProcessId(16777348) Memory Usage(17%) Clients(25) Order(4)
ProcessId(16777358) Memory Usage(17%) Clients(25) Order(5)
ProcessId(16777368) Memory Usage(17%) Clients(25) Order(6)
ProcessId(16777378) Memory Usage(17%) Clients(25) Order(7)
ProcessId(16777388) Memory Usage(17%) Clients(25) Order(8)
ProcessId(16777398) Memory Usage(17%) Clients(25) Order(9)
ProcessId(33554622) Memory Usage(17%) Clients(25) Order(10)
ProcessId(16777416) Memory Usage(17%) Clients(25) Order(11)
ProcessId(16777426) Memory Usage(17%) Clients(25) Order(12)
ProcessId(16777436) Memory Usage(9%) Clients(25) Order(13)
ProcessId(16777446) Memory Usage(17%) Clients(25) Order(14)
ProcessId(16777456) Memory Usage(17%) Clients(25) Order(15)
ProcessId(16777466) Memory Usage(17%) Clients(25) Order(16)
ProcessId(16777476) Memory Usage(17%) Clients(25) Order(17)
ProcessId(16777487) Memory Usage(17%) Clients(25) Order(18)
ProcessId(16777497) Memory Usage(17%) Clients(25) Order(19)
ProcessId(16777507) Memory Usage(16%) Clients(24) Order(20)

```

```

F RSED,APPL=D P,D
BPXM023I (STCRSE)
ProcessId(83886168) ASId(0022) JobName(RSED857 ) Order(1)
PROCESS LIMITS:      CURRENT  HIGHWATER    LIMIT
  JAVA HEAP USAGE(%)    17        17        100
    CLIENTS              25        25         25
  MAXFILEPROC           365       366      64000
  MAXPROCUSER            44        44         80
  MAXTHREADS            310       311      1500
  MAXTHREADTASKS        311       311      1500

```

TASID	Jobname	Cpu time	Storage	EXCP
	-----	-----	-----	-----
	JMON	0.00	1780	73
	RSED	5.88	95.2M	41958
	RSED1	8.26	190.1M	58669
	RSED1	8.17	187.0M	58605
	RSED2	8.06	185.3M	58653
	RSED2	8.19	183.1M	60209
	RSED3	8.12	189.1M	58650
	RSED3	8.03	186.7M	58590
	RSED4	8.15	188.2M	58646
	RSED4	5.50	182.5M	58585
	RSED5	7.72	184.4M	58631
	RSED5	7.82	184.1M	58576
	RSED6	7.14	184.1M	58622
	RSED6	6.27	186.9M	58583
	RSED7	5.17	185.1M	58804
	RSED7	6.57	185.2M	58621
	RSED7	5.86	182.8M	58565
	RSED8	0.36	1560	2459
	RSED8	7.94	184.1M	58615
	RSED8	7.45	181.8M	58548
	RSED9	8.16	190.6M	58802
	RSED9	7.62	183.8M	58610
	RSED9	7.36	177.7M	57478

図 25. サンプル・セットアップのリソース使用量

第 6 章 パフォーマンスに関する考慮事項

z/OS は高度にカスタマイズ可能なオペレーティング・システムであり、システムの場合によっては小さな) 変更で全体のパフォーマンスに大きな影響を与えることができます。この章では、Developer for System z のパフォーマンスを向上させるために行うことができるいくつかの変更を中心に説明します。

システムのチューニングの詳細については、「MVS 初期設定およびチューニングガイド」(SA88-8563)、および「UNIX System Services 計画」(GA88-8639)を参照してください。

zFS ファイル・システムの使用

zFS (zSeries ファイル・システム) および HFS (階層ファイル・システム) は、どちらも z/OS UNIX 環境で利用できる UNIX ファイル・システムです。ただし、zFS には、以下のようなフィーチャーと利点があります。

- 多数のカスタマー環境で頻繁にアクセスおよび更新される、サイズが 8 K に近いファイルにアクセスする際のパフォーマンスの向上。これより小さいファイルのアクセス・パフォーマンスは、HFS のそれと同等です。
- 同じデータ・セット内のファイル・システムの読み取り専用クローン作成。クローン作成されたファイル・システムを各ユーザーが使用できるようにして、ファイル・システムの読み取り専用ポイント・イン・タイム・コピーを提供することができます。これはオプションのフィーチャーで、非シスプレックス環境でのみ使用できます。
- zFS は、戦略的な z/OS UNIX ファイル・システムです。HFS の機能は、既に固定化されており、ファイル・システムの機能拡張は zFS だけのものです。

zFS の詳細については、「UNIX System Services 計画」(GA88-8639)を参照してください。

STEPLIB の使用の回避

親から子へ、または exec を越えて伝搬される STEPLIB を持つ z/OS UNIX プロセスは、それぞれが約 200 バイトの拡張共通ストレージ域 (ECSA) を消費します。STEPLIB 環境変数が定義されなかった場合、または STEPLIB=CURRENT として定義された場合、z/OS UNIX は現在アクティブなすべての TASKLIB、STEPLIB、および JOBLIB 割り振りを、fork()、spawn()、または exec() 関数の実行中に伝搬します。

Developer for System z では、rsed.envvars 構成ファイルの中で説明されているように、STEPLIB=NONE がデフォルトとして rsed.envvars 内にコーディングされています。上記の理由から、このディレクティブを変更するのではなく、代わりにターゲット・データ・セットを LINKLIST または LPA (リンク・パック域) の中に置いてください。

改善、システム・ライブラリーへのアクセスの

特定のシステム・ライブラリーとロード・モジュールは、z/OS UNIX およびアプリケーション開発アクティビティーによって頻繁に使用されます。それらをリンク・パック域 (LPA) に追加するなどの方法でアクセスを改善すると、システムのパフォーマンスを向上させることができます。以下で説明する SYS1.PARMLIB メンバーの変更について詳しくは、「MVS 初期設定およびチューニング解説書」(SA88-8564) を参照してください。

言語環境プログラム (LE) ランタイム・ライブラリー

C プログラム (z/OS UNIX シェルを含む) は、実行されると、言語環境プログラム (LE) ランタイム・ライブラリーからのルーチンを頻繁に使用します。平均すると、LE 対応プログラムを実行する 1 つのアドレス・スペースごとに約 4 MB のランタイム・ライブラリーがメモリーにロードされ、すべてのフォークにコピーされます。

CEE.SCEELPA

CEE.SCEELPA データ・セットには、z/OS UNIX によって頻繁に使用される LE ランタイム・ルーチンのサブセットが入っています。最大のパフォーマンス向上を実現するために、このデータ・セットを SYS1.PARMLIB(LPALSTxx) に追加する必要があります。そうすることにより、モジュールはディスクから 1 回だけ読み取られ、共用ケーションに保管されます。

注: ロード・モジュールを動的 LPA (リンク・パック域) に追加したい場合は、次のステートメントを SYS1.PARMLIB(PROGxx) に追加してください。

```
LPA ADD MASK(*) DSN(CEE.SCEELPA)
```

また、データ・セットを SYS1.PARMLIB(LNKLSTxx) または SYS1.PARMLIB(PROGxx) に追加することにより、LE ランタイム・ライブラリーを CEE.SCEERUN および CEE.SCEERUN2 に配置することもお勧めします。これにより、z/OS UNIX STEPLIB のオーバーヘッドがなくなり、LLA および VLF、またはそれらと同様な製品による管理のための入出力を削減できます。

注: 同じ理由から、C/C++ DLL クラス・ライブラリー CBC.SCLBDLL も、LINKLIST に追加してください。

これらのライブラリーを LINKLIST 内に置かない場合は、rsed.envvars 構成ファイルの中の説明に従い、適切な STEPLIB ステートメントを rsed.envvars 内にセットアップする必要があります。この方式では、常に追加の仮想ストレージが使用されますが、LE ランタイム・ライブラリーを LLA またはそれに類似した製品に対して定義することにより、パフォーマンスを改善できます。これにより、モジュールをロードするために必要な入出力が削減されます。

アプリケーション開発

アプリケーション開発が主なアクティビティーであるシステムでは、以下の行を SYS1.PARMLIB(PROGxx) に追加して、リンケージ・エディターを動的 LPA の中に置くと、パフォーマンスが向上する場合もあります。


```
LPA ADD MODNAME(CEEINIT,CEEBLIBM,CEEV003,EDCZV) DSNAME(CEE.SCEERUN)
LPA ADD MODNAME(IEFIB600,IEFXB603) DSNAME(SYS1.LINKLIB)
```

C/C++ 開発用に、CBC.SCCNCMP コンパイラー・データ・セットを
SYS1.PARMLIB(LPALSTxx) に追加することもできます。

上記のステートメントは、考えられる LPA 候補の例ですが、必要性はご使用のサイトによって異なる場合があります。他の LE ロード・モジュールを動的 LPA 内に配置する方法については、「*Language Environment カスタマイズ*」(SA88-8552)を参照してください。C/C++ コンパイラー・ロード・モジュールを動的 LPA の中に配置する方法については、「*UNIX System Services 計画*」(GA88-8639)を参照してください。

向上、セキュリティー検査のパフォーマンスの

z/OS UNIX について行われるセキュリティー検査のパフォーマンスを向上させるには、セキュリティー・ソフトウェアの FACILITY クラスで BPX.SAFFASTPATH プロファイルを定義します。これにより、さまざまな操作について z/OS UNIX セキュリティー検査を行うときのオーバーヘッドが削減されます。それらの検査には、ファイル・アクセス検査、IPC アクセス検査、およびプロセス所有権検査が含まれます。このプロファイルの詳細については、「*UNIX System Services 計画*」(GA88-8639)を参照してください。

注: ユーザーは、BPX.SAFFASTPATH プロファイルに対するアクセス権限を持っている必要はありません。

ワークロード管理

それぞれのサイトには固有の必要性があり、それを満たすために、z/OS オペレーティング・システムをカスタマイズして、使用可能なリソースを最大限に活用することができます。ワークロード管理を使用すると、パフォーマンスの最終目標を定義し、各目標にビジネスの重要度を割り当てることができます。作業の最終目標をビジネス用語で定義し、システムでは、その目標を達成するために、CPU やストレージなどのリソースをどれくらい作業に割り当てればよいかを決定します。

Developer for System z のパフォーマンスは、プロセスに正しい目標を設定することによってバランスをとることができます。以下に、いくつかの一般ガイドラインを示します。

- APPC トランザクションを使用する場合は、それを TSO パフォーマンス・グループに割り当てます。
- 開始タスク・パフォーマンス・グループ (SYSSTC) を Developer for System z サーバー・アドレス・スペース (JES ジョブ・モニター (JMON)、RSE デーモン (RSED)、および RSE スレッド・プール (RSEDx)) に割り当てます。

この件について詳しくは、「*MVS 計画: ワークロード管理*」(SA88-8574)を参照してください。

固定 Java ヒープ・サイズ

固定サイズのヒープを使用すると、ヒープの拡張や縮小が発生しないため、状態によっては、パフォーマンスの大幅な向上につながります。ただし、固定サイズのヒープを使用することは、通常ではお勧めできません。その理由は、ヒープが満杯になるまでガーベッジ・コレクションの開始が遅延し、開始された時点では、それがメジャー・タスクになるからです。また、フラグメント化の危険も増し、ヒープの圧縮が必要になります。したがって、固定サイズのヒープは、適正なテストの後か、IBM サポートからの指示の下でのみ使用してください。ヒープ・サイズとガーベッジ・コレクションの詳細については、「*Java Diagnostics Guide*」(SC34-6650)を参照してください。

z/OS Java 仮想マシン (JVM) の初期ヒープ・サイズおよび最大ヒープ・サイズは、Java コマンド行オプションの `-Xms` (初期) および `-Xmx` (最大) で設定できます。

Developer for System z では、Java コマンド行オプションは、「*ホスト構成ガイド*」(SC88-5663) の「『`_RSE_JAVAOPTS`」での追加 Java 始動パラメーター」の説明のように、`rsed.envvars` の `_RSE_JAVAOPTS` ディレクティブで定義されます。

```
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xms128m -Xmx128m"
```

Java -Xquickstart オプション

注: Java -Xquickstart は、RSE サーバーに REXEC/SSH 代替始動方式を使用する場合にのみ役立ちます。この方式については、「*ホスト構成ガイド*」(SC88-5663) の「『(オプション) REXEC (または SSH) の使用』」に説明があります。

-Xquickstart オプションは、一部の Java アプリケーションの起動時間を改善するために使用できます。-Xquickstart を指定すると、JIT (Just In Time) コンパイラーが最適化のサブセット、つまりクイック・コンパイルを使用して実行されます。このクイック・コンパイルでは、起動時間を改善できます。

-Xquickstart オプションは、実行期間が短いアプリケーション、特に実行時間が少数のメソッドに集中していないアプリケーションに適しています。-Xquickstart は、ホット・メソッドを含んでいる実行期間が長いアプリケーションに対して使用すると、パフォーマンスを低下させる場合があります。

RSE サーバー用に -Xquickstart オプションを有効にするには、次のディレクティブを `rsed.envvars` の末尾に追加します。

```
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xquickstart"
```

JVM 間でのクラス共有

IBM Java 仮想マシン (JVM) バージョン 5 以上では、ブートストラップ・クラスおよびアプリケーション・クラスを共有メモリー内のキャッシュに保管することにより、それらを JVM 間で共有できます。クラス共有は、複数の JVM がキャッシュを共有している場合、全体的な仮想メモリーの消費を削減します。また、クラス共有はキャッシュが作成された後、JVM の起動時間を短縮します。

共用クラス・キャッシュはアクティブな JVM に依存せず、キャッシュを作成した JVM の存続期間を越えて持続します。共用クラス・キャッシュは JVM の存続期間を越えて持続するので、JAR またはファイル・システム上のクラスに加えられたすべての変更が反映されるよう、キャッシュは動的に更新されます。

新しいキャッシュの作成とデータの取り込みを行うためのオーバーヘッドは最小限で済みます。時間的な JVM 始動コストは、単一の JVM の場合、ロードされるクラスの数によって異なりますが、クラス共用を使用しないシステムに比べて 0 から 5 % の低下が一般的です。キャッシュにデータが取り込まれた場合の JVM 起動時間の改善は、オペレーティング・システムおよびロードされるクラス数によって異なりますが、クラス共用を使用しないシステムに比べて 10 % から 40 % の高速化となって現れます。複数の JVM を並行して実行すると、全体的な起動時間がさらに改善されます。

クラス共用の詳細については、「*Java SDK and Runtime Environment User Guide*」を参照してください。

クラス共用の有効化

RSE サーバー用にクラス共用を有効にするには、次のディレクティブを `rse.envvars` の末尾に追加します。最初のステートメントは、グループ・アクセス権を持つ RSE という名前のキャッシュを定義しており、これは、クラス共用が失敗した場合でも RSE サーバーを始動できるようにします。2 番目のステートメントはオプションであり、キャッシュ・サイズを 6 メガバイトに設定します (システム・デフォルトは 16 MB です)。3 番目のステートメントは、クラス共用パラメーターを Java 始動オプションに追加します。

```
_RSE_CLASS_OPTS=-Xshareclasses:name=RSE,groupAccess,nonFatal
# _RSE_CLASS_OPTS="$ _RSE_CLASS_OPTS -Xscmx6m
_RSE_JAVA_OPTS="$ _RSE_JAVA_OPTS $ _RSE_CLASS_OPTS"
```

注: 『キャッシュ・セキュリティ』 で述べたように、共用クラスを使用するすべてのユーザーは、同じ 1 次グループ ID (GID) を持っている必要があります。つまり、それらのユーザーにはセキュリティ・ソフトウェア内で同じデフォルト・グループが定義されている必要があるか、さまざまなデフォルト・グループが OMVS セグメント内に同じ GID を持っています。

キャッシュ・サイズ制限

理論上の最大共用キャッシュ・サイズは 2 GB です。指定できるキャッシュのサイズは、システムで利用できる物理メモリーとスワップ・スペースの容量によって制限されます。プロセスの仮想アドレス・スペースは共用クラス・キャッシュと Java ヒープの間で共用されるので、Java ヒープの最大サイズを大きくすると、作成できる共用クラス・キャッシュのサイズが小さくなります。

キャッシュ・セキュリティ

共用クラス・キャッシュへのアクセスは、オペレーティング・システムの許可および Java セキュリティ許可によって制限されます。

デフォルトでは、クラス・キャッシュはユーザー・レベル・セキュリティを使用して作成されるため、キャッシュを作成したユーザーだけがキャッシュにアクセスできます。z/OS UNIX では、`groupAccess` というオプションがあり、これはキャッ

シュを作成したユーザーの 1 次グループに属するすべてのユーザーにアクセス権を与えます。しかし、使用されたアクセス・レベルに関係なく、キャッシュを破棄できるのは、そのキャッシュを作成したユーザーカルート・ユーザー (UID 0) だけです。

Java SecurityManager を使用した追加セキュリティー・オプションの詳細については、「*Java SDK and Runtime Environment User Guide*」を参照してください。

SYS1.PARMLIB(BPXPRMxx)

いくつかの SYS1.PARMLIB(BPXPRMxx) の設定は、共用クラスのパフォーマンスに影響します。誤った設定を使用すると、共用クラスが機能しなくなる可能性があります。また、それらの設定は、パフォーマンスへの影響を持つ場合もあります。これらのパラメーターのパフォーマンスへの影響と使用方法について詳しくは、「*MVS 初期設定およびチューニング解説書*」(SA88-8564) および「*UNIX System Services 計画*」(GA88-8639) を参照してください。共用クラスの操作に最も大きな影響を与える BPXPRMxx パラメーターは、以下のとおりです。

- MAXSHAREPAGES、IPCSHMSPAGES、IPCSHMMPAGES、および IPCSHMNSEGS

これらの設定は、JVM で使用できる共用メモリー・ページの量に影響します。31 ビット z/OS UNIX システム・サービスの共用ページ・サイズは、4 KB に固定されています。共用クラスは、デフォルトでは 16 MB のキャッシュを作成しようとしています。したがって、IPCSHMMPAGES は 4096 より大きく設定してください。

-Xscmx を使用してキャッシュ・サイズを設定する場合、JVM は最も近いメガバイト値まで切り上げを行います。使用するシステム上で IPCSHMMPAGES を設定するときは、このことを考慮する必要があります。

- IPCSEMNIIDS および IPCSEMNSEMS

これらの設定は、UNIX プロセスで使用できるセマフォの量に影響します。共用クラスは、IPC セマフォを使用して JVM 間の通信を行います。

ディスク・スペース

共用クラス・キャッシュは、システム上に存在するキャッシュの識別情報を保管するために、ディスク・スペースを必要とします。この情報は /tmp/javasharedresources に保管されます。識別情報ディレクトリーが削除された場合、JVM はシステム上の共用クラスを識別できないため、キャッシュの再作成が必要になります。

キャッシュ管理ユーティリティー

Java -Xshareclasses 行コマンドは、いくつかのオプションをとることができ、それらのオプションの一部はキャッシュ管理ユーティリティーです。そのうちの 3 つを以下のサンプルに示します (\$ は z/OS UNIX プロンプトです)。サポートされるコマンド行オプションの詳細については、「*Java SDK and Runtime Environment User Guide*」を参照してください。

```
$ java -Xshareclasses:listAllCaches
Shared Cache      OS shmid          in use            Last detach time
RSE               401412            0                 Mon Jun 18 17:23:16 2007
```

Could not create the Java virtual machine.

```
$ java -Xshareclasses:name=RSE,printStats
```

Current statistics for cache "RSE":

```
base address      = 0x0F300058
end address       = 0x0F8FFFF8
allocation pointer = 0x0F4D2E28
```

```
cache size        = 6291368
free bytes        = 4355696
ROMClass bytes    = 1912272
Metadata bytes    = 23400
Metadata % used   = 1%
```

```
# ROMClasses      = 475
# Classpaths      = 4
# URLs            = 0
# Tokens          = 0
# Stale classes   = 0
% Stale classes   = 0%
```

Cache is 30% full

Could not create the Java virtual machine.

```
$ java -Xshareclasses:name=RSE,destroy
JVMSHRC010I Shared Cache "RSE" is destroyed
Could not create the Java virtual machine.
```

注:

- キャッシュ・ユーティリティーは、指定されたキャッシュに対して必要な操作を、JVM を始動せずに実行するので、「Could not create the Java virtual machine」というメッセージは正常です。
- キャッシュを破棄できるのは、そのキャッシュを使用しているすべての JVM のシャットダウンが完了しており、しかもコマンドを発行するユーザーが十分な権限を持っている場合だけです。

第 7 章 クライアントへのプッシュの考慮事項

クライアントへのプッシュ (ホスト・ベースのクライアント制御) では、以下の事項についての集中管理をサポートしています。

- クライアントの構成ファイル
- クライアントの製品バージョン
- プロジェクト定義

この章では、以下のトピックについて説明します。

- 『概要』
- 132 ページの『1 次システム』
- 133 ページの『クライアントへのプッシュ・メタデータ』
- 135 ページの『クライアント構成の制御』
- 135 ページの『クライアント・バージョンの制御』
- 136 ページの『複数の開発者グループ』
- 140 ページの『LDAP ベースのグループ選択』
- 145 ページの『SAF ベースのグループ選択』
- 148 ページの『ホスト・ベースのプロジェクト』

概要

Developer for System z クライアント・バージョン 8.0.1 以上は、接続時にホストからクライアントの構成ファイルと製品の更新情報を取り出すことができるので、すべてのクライアントの設定が共通になり、最新のものになります。

バージョン 8.0.3 以降、クライアント管理者は、異なる開発者グループのニーズに適合するように、複数のクライアント構成のセットと複数のクライアント更新シナリオを作成できるようになりました。これにより、ユーザーは、LDAP グループのメンバーシップやセキュリティ・プロファイルに対する許可などの基準に基づいてカスタマイズされたセットアップを受け取れるようになります。

z/OS プロジェクトは、クライアント上の「z/OS プロジェクト」パースペクティブで個別に定義できます。または、z/OS プロジェクトをホスト上で集中的に定義してクライアントに対して個々のユーザー単位で伝搬することもできます。それらの「ホスト・ベースのプロジェクト」は、クライアント上で定義されたプロジェクトと外観も機能もまったく同じですが、クライアントは、それらの構造、メンバー、およびプロパティーを変更できず、ホストに接続している場合にのみ、それらのプロジェクトにアクセスできます。

`pushtoclient.properties` はクライアントに対し、これらの機能が使用可能であるか、関連のデータがどこに保管されているかを伝達します。詳しくは、「ホスト構成ガイド」(SC88-5663) の『(オプション) `pushtoclient.properties`、ホスト・ベースのクライアント制御』を参照してください。

一般的に、z/OS システム、開発者のワークステーション、および開発プロジェクトは、それぞれに異なるメンバーのグループが管理しています。クライアントへのプッシュの設計は、この原則に従って各グループに特定の役割を割り当てます。

- z/OS システム・プログラマーは、クライアントへのプッシュ・メタデータのローテーション、基本的なセキュリティの側面、およびクライアントへのプッシュをアクティブにするかどうかを制御します。
- クライアント管理者は、クライアントへのプッシュ・メタデータの内容を保守します。そのために、Developer for System z クライアントを使用して 1 つ以上のクライアントの構成を作成し、IBM Installation Manager を使用して Developer for System z クライアントの更新に使用する応答ファイルを作成します。
- 開発プロジェクト・マネージャーは、プロジェクトを定義して個々の開発者にそのプロジェクトを割り当てます。

クライアント管理者と開発プロジェクト・マネージャーがそれぞれに割り当てられたタスクを実行する方法については、Developer for System z インフォメーション・センター (<http://pic.dhe.ibm.com/infocenter/ratdevz/v9r0/index.jsp>) を参照してください。

複数の開発者グループに対する構成またはバージョン制御のサポートを使用可能にするときには、クライアントへのプッシュの管理に 1 つの追加のチームが関与することになります。これがどのチームになるかは、開発者が所属するグループを識別するために選択したオプションに応じて異なります。

- LDAP 管理者は、各開発者を 1 つまたは複数の FEK.PTC.* LDAP グループに配属する、または配属しないようにするグループ定義を保守します。
- セキュリティー管理者は、FEK.PTC.* セキュリティー・プロファイルに対するアクセス・リストを保守します。開発者には、1 つまたは複数のプロファイルに対する許可を与えることも、プロファイルに対する許可を与えないこともできます。

1 次システム

クライアントへのプッシュは、システム固有のデータをシステムごとに保管し、一方で共通の (全体的な) データは単一のシステム (1 次システム) 上で維持して管理上の負担を軽減するように設計されています。1 次システムは、`pushtoclient.properties` の `primary.system` ディレクティブで識別されます。デフォルトは `false` です。

1 つのシステムのみを 1 次システムとして定義するようにしてください。Developer for System z クライアント管理者は、ターゲット・システムが 1 次システムでなければ、グローバル構成データをエクスポートできなくなります。Developer for System z クライアントは、構成が同期していない複数の 1 次システムに接続する場合、不規則な動作を示すことがあります。

この唯一の規則は、複数のシステムが Developer for System z 構成 (`/etc/rdz`) およびクライアントへのプッシュ・メタデータ (`/var/rdz/pushtoclient`) を共有する場合には適用されません。構成が共有されるため、関与するすべてのシステムは 1 次システムとして識別されます。ただし、すべてのシステムがメタデータも共有していれば、この重複が問題になることはありません。

クライアントへのプッシュ・メタデータ

メタデータのロケーション

クライアントへのプッシュ・メタデータが格納されるベース・ディレクトリーは、`pushtoclient.properties` の `pushtoclient.folder` ディレクティブで識別します。デフォルトは `/var/rdz/pushtoclient` です。

ベース・ディレクトリーには、ルートのクライアントへのプッシュ構成ファイルである `keymapping.xml` が保持されます。その他すべてのメタデータは、サブディレクトリーに保持されます。

ほとんどのサブディレクトリーは、クライアント管理者がクライアントへのプッシュ・ワークスペース構成をエクスポートするときに動的に作成されます。これらのサブディレクトリーによって、マッピングや設定などのサブジェクトごとにメタデータをグループ化します。クライアントへのプッシュによる管理に適した **Developer for System z** クライアント・コンポーネント数が増えるほど、より多くのサブディレクトリーが動的に作成されるようになります。これらのサブディレクトリーに何が格納されているのかを調べるには、**Developer for System z** クライアントでエクスポート・ウィザード (「ファイル」>「エクスポート...」>「**Rational Developer for System z**」>「構成ファイル」) を参照してください。

いくつかのサブディレクトリーは、ホストの初期カスタマイズ中に作成されます。これらのサブディレクトリーには、クライアント管理者または開発プロジェクト・マネージャーが手動で保守するデータが保持されます。

- `/var/rdz/pushtoclient/projects/` には、ホスト・ベースのプロジェクト定義ファイルが保持されます。実際のロケーションは、`/var/rdz/pushtoclient/keymapping.xml` で指定されます。これは、**Developer for System z** クライアント管理者が作成し、保守します。この中のファイルは、プロジェクト・マネージャーまたは主任開発者が保守します。
- `/var/rdz/pushtoclient/install/` には、ホストへの接続時にクライアント製品のバージョンを更新するために使用される構成ファイルが保持されます。実際のロケーションは、`/var/rdz/pushtoclient/keymapping.xml` で指定されます。これは、**Developer for System z** クライアント管理者が作成し、保守します。この中のファイルは、クライアント管理者が保守します。
- `/var/rdz/pushtoclient/install/responsefiles/` には、ホストへの接続時にクライアント製品のバージョンを更新するために使用される構成ファイルが保持されます。実際のロケーションは、`/var/rdz/pushtoclient/keymapping.xml` で指定されます。これは、**Developer for System z** クライアント管理者が作成し、保守します。この中のファイルは、クライアント管理者が保守します。

これらのサブディレクトリーの作成について詳しくは、「ホスト構成ガイド」(SC88-5663) の章『基本的なカスタマイズ』の『カスタマイズのセットアップ』を参照してください。

メタデータのセキュリティ

デフォルト (`pushtoclient.properties` の `file.permission` ディレクティブで確認できます) では、ベース・ディレクトリーに作成されるすべてのファイルとディレ

クトリーに対して許可ビット・マスク 775 (rwxrwxr-x) が設定されます。これにより、所有者と所有者のデフォルト・グループは、そのディレクトリー構造と、そこに含まれるファイルに対する読み取りアクセスと書き込みアクセスが許可されます。その他のユーザーは、そのディレクトリー構造と、そこに含まれるファイルに対する読み取り権限のみを持ちます。

クライアントへのプッシュのセットアップを開始する前に、これらのディレクトリーに適切な所有者 UID (ユーザー ID) と GID (グループ ID) を設定することが重要です。

以下の RACF コマンドの例では、新規グループ (RDZADMIN) を作成して固有の GID (2) を割り当て、そのグループをユーザー ID RDZADM1 のデフォルト・グループにします。さらに、RDZADM1 には固有の UID (6) を与えます。

```
ADDGROUP RDZADMIN OWNER(IBMUSER) SUPGROUP(SYS1) -  
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - CLIENT ADMIN')  
ALTGROUP RDZADMIN OMVS(GID(2))  
CONNECT RDZADM1 GROUP(RDZADMIN) AUTH(USE)  
ALTUSER RDZADM1 DFLTGRP(RDZADMIN) OMVS(UID(6))
```

以下の z/OS UNIX の **chown** コマンド例では、/var/rdz/pushtoclient の所有者とグループを変更して、そのディレクトリー内のすべての所有者とグループをそれぞれ RDZADM1 と RDZADMIN にします。このコマンドは、アクセス権の問題を避けるために、スーパーユーザー (UID 0) が実行する必要があります。

```
chown -R rdzadm1:rdzadmin /var/rdz/pushtoclient
```

以下の z/OS UNIX の **chmod** コマンド例では、/var/rdz/pushtoclient の許可ビット・マスクを変更して、そのディレクトリー内のすべての許可ビット・マスクを 775 にします。これを実行すると、このディレクトリーに対する手動の追加操作が Developer for System z で使用されるロジックに必ず従うようになります。このコマンドは、アクセス権の問題を避けるために、スーパーユーザー (UID 0) が実行する必要があります。

```
chmod -R 775 /var/rdz/pushtoclient
```

サンプル RACF コマンドの詳細については、「*Security Server RACF コマンド言語解説書*」(SA88-8617) を参照してください。サンプル z/OS UNIX コマンドについては、「*UNIX System Services コマンド解説書*」(SA88-8641) を参照してください。追加情報については、15 ページの『z/OS UNIX ディレクトリー構造』を参照してください。

メタデータのスペース使用量

クライアントへのプッシュ・メタデータは z/OS UNIX のディスク・スペースをそれほど大量には使用しません。これは、このメタデータの大部分が UTF-8 でエンコードされた XML ファイルであるためです。クライアントの更新シナリオに使用する製品コードは、ネットワーク上のどこにでも保管できることに注意してください。この製品コードを z/OS UNIX に保管する必要はありません。関連するクライアントへのプッシュ・メタデータ (応答ファイルと呼びます) によって、クライアントは正しいロケーションをポイントするようになります。

クライアント構成の制御

Developer for System z クライアント (バージョン 8.0.1 以降) は、ホストへの接続時に `pushtoclient.properties` にある定義を読み取ります。ディレクティブ `config.enabled` が有効にされている場合、クライアントは、そのクライアントの現行の構成とクライアントへのプッシュ・メタデータ内の定義を比較します。差異が検出されると、そのクライアントは必要なデータを取り出すウィザードを開始し、クライアントへのプッシュによる指示に従ってセットアップをアクティブ化します。

`pushtoclient.properties` の `reject.config.updates` ディレクティブでは、クライアントへのプッシュによって送信される構成の更新をユーザーが拒否できるようにするかどうかを制御します。

Developer for System z クライアント (バージョン 8.0.1 以降) には、クライアント管理者が使用するウィザードがあります。このウィザードを使用すると、現行の構成をエクスポートできます。このエクスポートした構成は、クライアントへのプッシュによってすべての Developer for System z クライアントにインポートできます。この機能は、すべてのクライアントで使用可能になることに注意してください。そのため、クライアントへのプッシュ・メタデータを保持する z/OS UNIX のディレクトリー (`/var/rdz/pushtoclient`) に対する書き込みアクセス権は、必ずクライアント管理者にのみ付与してください。

グループ・サポートを使用可能にするには、クライアントとホストの両方がバージョン 8.0.3 以降である必要があります。詳しくは、136 ページの『複数の開発者グループ』を参照してください。

クライアント・バージョンの制御

Developer for System z クライアント (バージョン 8.0.1 以降) は、ホストへの接続時に `pushtoclient.properties` にある定義を読み取ります。ディレクティブ `product.enabled` が有効にされている場合、クライアントは、そのクライアントの現行の製品バージョンとクライアントへのプッシュ・メタデータ内の定義を比較します。差異が検出されると、そのクライアントは必要なデータを取り出すウィザードを開始し、クライアントへのプッシュによる指示に従ってセットアップをアクティブ化します。

`pushtoclient.properties` の `reject.product.updates` ディレクティブでは、クライアントへのプッシュによって送信される製品の更新をユーザーが拒否できるようにするかどうかを制御します。

グループ・サポートを使用可能にするには、クライアントとホストの両方がバージョン 8.0.3 以降である必要があります。詳しくは、136 ページの『複数の開発者グループ』を参照してください。

複数の開発者グループ

バージョン 8.0.3 以降、クライアント管理者は、異なる開発者グループのニーズに適合するように、複数のクライアント構成のセットと複数のクライアント更新シナリオを作成できるようになりました。これにより、ユーザーは、LDAP グループのメンバーシップやセキュリティ・プロファイルに対する許可などの基準に基づいてカスタマイズされたセットアップを受け取れるようになります。

アクティベーション

各グループが独自のクライアント構成とクライアント更新の要件を持つ複数の開発者グループに対するサポートは、表 33 に示されているように、`pushtoclient.properties` に含まれる関連ディレクティブ (`config.enabled` および `product.enabled`) に適切な値を割り当てることで使用可能になります。

表 33. `*.enabled` に関するクライアントへのプッシュのグループ・サポート・マトリックス

<code>*.enabled</code> の値	機能の有効化	複数のグループのサポート
False	いいえ	いいえ
True	はい	いいえ
LDAP	はい	はい。LDAP グループ FEK.PTC.*.ENABLED.sysname.devgroup のメンバーシップに基づきます。
SAF	はい	はい。セキュリティ・プロファイル FEK.PTC.*.ENABLED.sysname.devgroup に基づきます。

この機能を有効化すると (これには TRUE 値が含まれます)、開発者は必ずデフォルト・グループに所属することになるため、注意が必要です。開発者は、1 つまたは複数の追加グループに所属することも、追加グループに所属しないこともできます。

表 34 に示すように、条件付きで更新を拒否することもできます。

表 34. `reject.*.updates` に関するクライアントへのプッシュのグループ・サポート・マトリックス

<code>reject.*.updates</code> の値	機能の有効化
False	いいえ
True	はい
LDAP	LDAP グループ・メンバーシップ FEK.PTC.REJECT.*.UPDATES.sysname に依存します。
SAF	セキュリティ・プロファイル FEK.PTC.REJECT.*.UPDATES.sysname に対する許可に依存します。

`pushtoclient.properties` のディレクティブは相互に独立して機能することに注意してください。サポートされている値は、どのディレクティブにでも割り当てることができます。設定を一様に保つ必要はありません。

それぞれの機能に必要なセットアップについて詳しくは、140 ページの『LDAP ベースのグループ選択』および 145 ページの『SAF ベースのグループ選択』を参照してください。複数グループのサポートを有効にする方法についての詳細は、「ホ

スト構成ガイド」(SC88-5663) の『(オプション) pushtoclient.properties、ホスト・ベースのクライアント制御』を参照してください。

グループの連結

pushtoclient.properties の *.enabled 機能が有効な場合 (TRUE 値を含む)、開発者は関連する機能のデフォルト・グループに必ず所属しています。開発者は、1 つまたは複数の追加グループに所属することも、追加グループに所属しないこともできます。

複数のグループで定義された変更を適用する際の複雑性を制限するために、Developer for System z では、ユーザーが行った選択に基づいて、使用される定義を制限します。

表 35. クライアントへのプッシュ・グループの連結

追加グループ	使用する定義
なし	デフォルト
1 つ	デフォルト、または (デフォルト + グループ)
複数	デフォルト、または (デフォルト + 1 グループ)

Developer for System z では、変更セットの作成および適用の際には、以下のロジックを使用します。

1. デフォルト定義で指定された更新があれば、その更新を取得する。
2. 選択したグループ定義で指定された更新があれば、その更新を取得する。既にデフォルトの更新が存在する場合は、デフォルトの更新を変更する。
3. クライアントに更新を適用する。

注: 更新は、削除アクション、追加アクション、およびオーバーレイ・アクションで構成できます。

ワークスペースのバインディング

開発者が複数のグループに同時に所属することはできますが、開発者のアクティブ・ワークスペースを複数のグループにバインドすることはできません。アクティブ・ワークスペースは、構成の更新または製品の更新を受信するために、特定のグループにバインドする必要があります (これは、デフォルト・グループでも構いません)。完了したバインドは、元に戻すことができません。新規のグループ・バインディングが必要な場合には、新規のワークスペースを作成する必要があります。

グループ・バインディングのないワークスペースがホストに接続するときに、config.enabled (または product.enabled) でそのワークスペースのクライアントへのプッシュがアクティブであることが示されていると、Developer for System z はユーザーが所属しているグループを特定するためにすべてのグループを照会して、関連する機能のグループを選択するように促すプロンプトをユーザーに表示します。それ以降の接続時には、グループ・メンバーシップが依然として有効であるかどうかを確認するために、選択したグループのみが照会されます。

reject.*.updates ディレクティブは複数のグループに対しては機能しないので、それらのセットアップはより単純になり、ワークスペース・バインディングも必要あ

りません。更新が提示されると、Developer for System z はユーザーが更新の拒否を許可されているかどうかを判別して、それに応じた動作を実行します。

グループ・メタデータのロケーション

133 ページの『メタデータのロケーション』で説明されているように、グループ・サポートなしのセットアップを使用すると、クライアントへのプッシュ・メタデータはすべて、`/var/rdz/pushtoclient/` の最上位のディレクトリー構造に格納されます。グループ・サポートがアクティブにされていても同一のレイアウトが維持されますが、ベース・ディレクトリー `/var/rdz/pushtoclient/` の解釈はわずかに異なります。

- `/var/rdz/pushtoclient/` 内の既存のデータは、デフォルト・グループのデータとして解釈されます。デフォルト・グループに向けてエクスポートすると、`/var/rdz/pushtoclient/` 内にメタデータが作成されるか、このディレクトリー内のメタデータが更新されます。このように解釈することで、バージョン 8.0.1 のクライアントや、バージョン 8.0.2 のクライアントとの互換性が保証されます。これらのバージョンのクライアントでは、クライアントへのプッシュは使用可能ですが、複数のグループをサポートしていません。
- 特定のグループに向けてエクスポートすると、`/var/rdz/pushtoclient/` の代わりに `/var/rdz/pushtoclient/grouping/<devgroup>/` がベース・ディレクトリーとして扱われ、このディレクトリー内にメタデータが作成されるか、このディレクトリー内のメタデータが更新されます。<devgroup> 値は、特定の開発者グループに割り当てられたグループ名と一致します。

製品の初期カスタマイズでは、`/var/rdz/pushtoclient/` 内に `grouping/` ディレクトリーが作成されます。`/var/rdz/pushtoclient/grouping/` に `<devgroup>/` ディレクトリーを追加することは、クライアント管理者の責任です。

製品の初期カスタマイズ中に、`projects/`、`install/` および `install/responsefiles/` の各ディレクトリーが `/var/rdz/pushtoclient/` 内に作成されることに注意してください。クライアント管理者は、`/var/rdz/pushtoclient/grouping/<devgroup>/` 内で、これらのディレクトリー作成操作を繰り返す必要があります（グループ固有の製品アップグレード・シナリオやグループ固有のホスト・ベースのプロジェクトに必要な場合）。

以下に示す z/OS UNIX コマンドのサンプル・シーケンスでは、適切な権限ビット・マスクを使用してサブディレクトリーを作成します。このコマンドは、所有権の問題を避けるために、クライアント管理者が実行する必要があります。

```
saved_umask=$(umask)
umask 0000
cd /var/rdz/pushtoclient/grouping/
mkdir -m775 <devgroup>
cd <devgroup>
mkdir -m775 install
mkdir -m775 install/responsefiles
mkdir -m775 projects
umask $saved_umask
```

サンプル z/OS UNIX コマンドについて詳しくは、「UNIX System Services コマンド解説書」(SA88-8641) を参照してください。

セットアップ手順

複数の開発者グループのサポートをセットアップするには、z/OS システム・プログラマー、クライアント管理者および選択基準を管理する管理者 (LDAP 管理者またはセキュリティ管理者) の間での調整が必要になります。以下に示すワークフローの説明では、セキュリティ管理者が選択基準を管理します。

1. クライアント管理者は、セキュリティ管理者に対して既存の開発者のグループ化セットアップについての入力を依頼します。既存のセットアップを再利用することで、クライアントへのプッシュのセットアップを短縮および単純化できます。
2. クライアント管理者は、複数グループのサポートを構造化する方法と、それらのクライアントへのプッシュ・グループに所属させる必要のあるメンバーを決定します。

注:

- デフォルトの構成セットとデフォルトの製品更新シナリオは常に存在します。
 - クライアントへのプッシュ変更セットには、削除アクション、追加アクション、およびオーバーレイ・アクションを含めることができます。
 - クライアントへのプッシュ変更セットは、空にすることができます。
 - 開発者は、1 つまたは複数のクライアントへのプッシュ・グループに所属することも、所属しないこともできます。
 - クライアント管理者は、各クライアントへのプッシュ・グループのメンバーである必要があります。
3. クライアント管理者とセキュリティ管理者は、クライアントへのプッシュ・グループに使用する名前について同意する必要があります。
 4. クライアント管理者は

```
/var/rdz/pushtoclient/grouping/<devgroup>
```

ディレクトリーをクライアントへのプッシュ・グループごとに作成します。

注: このディレクトリーの許可ビットは、775 (drwxrwxr-x) である必要があります。

5. セキュリティ管理者は、必要な初期セットアップを行ってクライアントへのプッシュの選択基準プロファイルを定義し、クライアントへのプッシュ・グループをアクセス・リストに追加します。

注:

- アクセス・リストに記載されたクライアント管理者が、関連するクライアントへのプッシュ・メタデータを作成するためには、少なくともそのクライアント管理者と一緒に選択基準の構造を定義しておく必要があります。
 - 初期セットアップでは、クライアントへのプッシュ・グループのアクセス・リストにはクライアント管理者だけを含めるようにしてください。これは、Developer for System z クライアントが、準備中のセットアップを受信しないようにするためです。
6. z/OS システム・プログラマーは、pushtoclient.properties を調整することで、複数グループのサポートをアクティブ化します。

注: *.enabled ディレクティブは、関連するクライアントへのプッシュ・メタデータを作成する前にクライアント管理者が有効にする必要があります。

7. クライアント管理者は、各グループのワークスペースを作成してから、それぞれのグループ名を使用してホストにエクスポートします。また、クライアント管理者はグループ固有の製品更新シナリオの作成に必要な応答ファイルも作成します。
8. セキュリティ管理者は、クライアントへのプッシュ・グループに開発者を追加し、開発者に対してクライアントへのプッシュをアクティブにします。

LDAP ベースのグループ選択

LDAP (Lightweight Directory Access Protocol) は TCP/IP ベースのプロトコルの名前ですが、分散ディレクトリー・サービスのセットを表現するために一般に使用されています。データベースと同様に、ディレクトリーは構造化されたレコードのコレクションです。Developer for System z では、各グループが 1 人以上のメンバーを保持する単純な階層データベースとして、LDAP サーバーを使用できます。

LDAP サーバーの定義を選択手段として使用する場合 (pushtoclient.properties のディレクティブに LDAP 値が指定されている場合)、Developer for System z は表 36 にリストされているグループ名のメンバーシップを検証して、ユーザーが所属している開発者グループを判別したり、ユーザーが更新の拒否を許可されているかどうかを判別します。

表 36. クライアントへのプッシュの LDAP 情報

グループ名 (cn=)	結果
FEK.PTC.CONFIG.ENABLED.sysname.devgroup	クライアントは指定されたグループ用の構成の更新を受諾する
FEK.PTC.PRODUCT.ENABLED.sysname.devgroup	クライアントは指定されたグループ用の製品の更新を受諾する
FEK.PTC.REJECT.CONFIG.UPDATES.sysname	ユーザーは構成の更新を拒否できる
FEK.PTC.REJECT.PRODUCT.UPDATES.sysname	ユーザーは製品の更新を拒否できる

devgroup 値は、特定の開発者グループに割り当てられたグループ名と一致します。グループ名は、Developer for System z クライアントに表示されることに注意してください。

sysname の値は、ターゲット・システムのシステム名と一致します。

LDAP スキーマ

LDAP スキーマは以下の規則を満たす必要があります。

1. 各クライアントへのプッシュ・グループは、スキーマ内でグループとして定義されている必要があります。
2. 各ユーザーは、スキーマ内でユーザーとして定義されている必要があります。
3. グループ項目には、そのグループ自体に所属するユーザー項目への参照があります。

図 26 は、LDIF 形式で表現されたグループとユーザーの LDAP 定義の例です。

注: LDAP データ交換形式 (LDIF) は、LDAP オブジェクトおよび LDAP 更新を表現するための標準テキスト形式です。LDIF レコードを含むファイルは、ディレクトリー・サーバー間でのデータ転送に使用することも、LDAP ユーティリティの入力として使用することもあります。

```
# Group Definition
dn: cn=FEK.PTC.CONFIG.ENABLED.CDFMVS08.GROUPA,o=PTC,c=DeveloperForZ
objectClass: groupOfUniqueNames
objectClass: top
cn: FEK.PTC.CONFIG.ENABLED.CDFMVS08.GROUPA
description: Project A
uniqueMember: uid=mborn,ou=Users,dc=example,dc=com

# User Definition
dn: uid=mborn,ou=Users,dc=example,dc=com
objectClass: organizationalPerson
objectClass: person
objectClass: inetOrgPerson
objectClass: uidObject
objectClass: top
cn: May Born
sn: Born
uid: mborn
facsimiletelephonenumber: +1 800 982 6883
givenname: May
mail: mborn@example.com
ou: Users
```

図 26. LDAP スキーマ定義の例

LDAP サーバーの選択

使用可能な商用 LDAP サーバーおよびフリーの LDAP サーバーには、数多くの選択肢があります。その一例として、IBM Tivoli® Directory Server (<http://www-01.ibm.com/software/tivoli/products/directory-server/>) があります。また、LDAP サーバーを管理するためのコマンド行ツールおよび GUI ベースのツールにも、数多くの選択肢があります。

140 ページの『LDAP スキーマ』で説明されているように、各ユーザーは LDAP サーバーに定義されている必要があります。管理作業を軽減するための最善策は、すべてのユーザー定義にアクセス可能になっている LDAP サーバーで、クライアントへのプッシュ・スキーマを実施することです。例えば、SDBM データベース (セキュリティ・データベースのラッパー) を使用する z/OS 上でアクティブな IBM Tivoli Directory Server が使用できます。

サイト・ポリシーによっては、LDAP サーバーでのクライアントへのプッシュ・スキーマをクライアント管理者が管理することになります。このような配置は、コラボレーションの必要性と、遅延や通信エラーの可能性を減少できます。

クライアント管理者による LDAP 管理が推奨される理由は、クライアントへのプッシュ・スキーマが機密情報やセキュリティ関連の情報を一切保持しないためです。その他のスキーマによって LDAP サーバーがユーザー定義を使用できる場合に

は、Developer for System z LDAP オブジェクトは、開発者が選択しているワークスペースのレイアウトと Developer for System z クライアント製品の自動更新を判断するだけになります。

LDAP サーバーのロケーション

LDAP プロトコルをサポートする任意のデータベース・サーバーを、Developer for System z のクライアントへのプッシュ・スキーマのホストに使用できます。そのため、Developer for System z では、LDAP サーバーへの接続に必要な情報を指定できます。また、LDAP サーバー内でデータベースを一意に特定するサフィックスを指定することもできます。

rsed.envvars ディレクティブ	デフォルト
_RSE_LDAP_SERVER	ローカル・ホスト・システム
_RSE_LDAP_PORT	389
_RSE_LDAP_PTC_GROUP_SUFFIX	"O=PTC,C=DeveloperForZ"

ファイアウォールなどの TCP/IP セキュリティ保護手段によって、(ホストベースの) RSE サーバーが LDAP サーバーと通信できなくなることがある点に注意してください。TCP/IP 管理者に連絡を取り、以下の情報を伝えて確実に LDAP サーバーにアクセスできるようにしてください。

- LDAP サーバーの TCP/IP アドレスまたは DNS 名
- LDAP サーバーのポート番号
- LDAP は TCP プロトコルを使用する
- LDAP サーバーは、ホストベースの RSE サーバーによってアクセスされる
- RSE サーバーは、RSEDx アドレス・スペース内でアクティブである (RSED は RSE 開始タスクの名前になります。x はランダムな 1 桁の数値になります)

サンプル・セットアップ

ある企業がシステム CDFMVS08 上で Developer for System z をアクティブにしていると仮定します。また、LDAP サーバーとして使用される IBM Tivoli Directory Server も CDFMVS08 上でアクティブだと仮定します。この LDAP サーバーは、143 ページの『クライアントへのプッシュ・バックエンドの LDAP への追加』で説明されているように構成されています。

Developer for System z を使用するユーザーは、以下のとおりです。

- 銀行アプリケーションに関する作業を行う複数の開発者。ユーザー ID BNK010 -> BNK014
- 保険アプリケーションに関する作業を行う複数の開発者。ユーザー ID INS010 -> INS014
- Developer for System z クライアント管理者が 1 人。ユーザー ID RDZADM1

各開発者グループは特定のクライアント構成ファイルを必要とします。また、すべての開発者は同一のクライアント・バージョン制御の管理下にあります。クライアント管理者とは異なり、開発者はクライアントへのプッシュが提示する変更を拒否することは許可されません。

クライアント管理者と LDAP 管理者は、構成の更新にグループ名 BANKING および INSURANCE を使用することについて同意する必要があります。

クライアントへのプッシュ・バックエンドの LDAP への追加

この例では、現時点で SDBM データベース (セキュリティー・データベース・ラッパー) のみを使用している z/OS 上の IBM Tivoli Directory Server に、LDBM データベース (z/OS UNIX のファイル) を追加することで、クライアントへのプッシュ・スキーマをホストするように更新します。

1. LDBM バックエンド・セクションを LDAP 構成ファイルに追加します。

```
# filename ds.conf
# restart GLDSRV started task to pick up changes

# global section
adminDN "cn=LDAP admin"
adminPW password
listen ldap://:389
schemaPath /etc/ldap

# SDBM back-end section (RACF)
database SDBM GLDBSD31/GLDBSD64
suffix "cn=RACF,o=IBM,c=US"

# LDBM back-end section (z/OS UNIX files)
database LDBM GLDBLD31/GLDBLD64 LDBM-RDZ
suffix "o=PTC,c=DeveloperForZ"
databaseDirectory /var/ldap/ldbm/rdz
```

2. LDAP 開始タスク GRDSRV を停止してから開始して、構成の変更を取り込みます。
3. /var/ldap/ldbm/rdz ディレクトリを作成します。

```
mkdir -p /var/ldap/ldbm/rdz
```

4. LDAP スキーマを更新して、LDBM バックエンドを追加します。

```
ldapmodify -D "cn=LDAP admin" -w password -f
/usr/lpp/ldap/etc/schema.user.ldif

ldapmodify -D "cn=LDAP admin" -w password -f
/usr/lpp/ldap/etc/schema.IBM.ldif
```

5. ルート項目を LDBM バックエンドに追加します。

```
ldapadd -D "cn=LDAP admin" -w password -f
/u/ibmuser/ptc_root.ldif
```

上記の /u/ibmuser/ptc_root.ldif は、以下の項目を保持します。

```
dn: o=PTC,c=DeveloperForZ
objectclass: top
objectclass: organization
o: PTC
```

LDAP グループの初期セットアップ

スキーマに異なる LDAP グループ・オブジェクトを追加し、クライアント管理者をそれらの一部にします。ユーザー ID RDZADM1 のユーザー定義は、RACF スキーマから取り出します。

```
ldapadd -D "cn=LDAP admin" -w password -f /u/ibmuser/ptc_setup.ldif
```

上記の /u/ibmuser/ptc_setup.ldif は、以下の項目を保持します。

```
# banking workspace configuration
dn: cn=FEK.PTC.CONFIG.ENABLED.CDFMVS08.BANKING,o=PTC,c=DeveloperForZ
objectclass: groupOfUniqueNames
cn: FEK.PTC.CONFIG.ENABLED.CDFMVS08.BANKING
description: Developer for System z push-to-client
# give client administrator access
uniqueMember: racfID=RDZADM1,profileType=user,cn=RACF,o=IBM,c=US

# insurance workspace configuration
dn: cn=FEK.PTC.CONFIG.ENABLED.CDFMVS08.INSURANCE,o=PTC,c=DeveloperForZ
objectclass: groupOfUniqueNames
cn: FEK.PTC.CONFIG.ENABLED.CDFMVS08.INSURANCE
description: Developer for System z push-to-client
# give client administrator access
uniqueMember: racfID=RDZADM1,profileType=user,cn=RACF,o=IBM,c=US

# reject configuration updates
dn: cn=FEK.PTC.REJECT.CONFIG.UPDATES.CDFMVS08,o=PTC,c=DeveloperForZ
objectclass: groupOfUniqueNames
cn: FEK.PTC.REJECT.CONFIG.UPDATES.CDFMVS08
description: Developer for System z push-to-client
# give client administrator access
uniqueMember: racfID=RDZADM1,profileType=user,cn=RACF,o=IBM,c=US

# reject product updates
dn: cn=FEK.PTC.REJECT.PRODUCT.UPDATES.CDFMVS08,o=PTC,c=DeveloperForZ
objectclass: groupOfUniqueNames
cn: FEK.PTC.REJECT.PRODUCT.UPDATES.CDFMVS08
description: Developer for System z push-to-client
# give client administrator access
uniqueMember: racfID=RDZADM1,profileType=user,cn=RACF,o=IBM,c=US
```

LDAP グループへの開発者の追加

開発者を LDAP グループ・オブジェクトに追加します。ユーザー ID のユーザー定義は、RACF スキーマから取り出されます。

```
ldapmodify -D "cn=LDAP admin" -w password -f /u/ibmuser/ptc_add.ldif
```

ここでは、/u/ibmuser/ptc_add.ldif は以下の情報を保持します。

```
# banking workspace configuration
dn: cn=FEK.PTC.CONFIG.ENABLED.CDFMVS08.BANKING,o=PTC,c=DeveloperForZ
changeType: modify
add: uniqueMember
uniqueMember: racfID=BNK010,profileType=user,cn=RACF,o=IBM,c=US
uniqueMember: racfID=BNK011,profileType=user,cn=RACF,o=IBM,c=US
uniqueMember: racfID=BNK012,profileType=user,cn=RACF,o=IBM,c=US
uniqueMember: racfID=BNK013,profileType=user,cn=RACF,o=IBM,c=US
uniqueMember: racfID=BNK014,profileType=user,cn=RACF,o=IBM,c=US

# insurance workspace configuration
dn: cn=FEK.PTC.CONFIG.ENABLED.CDFMVS08.INSURANCE,o=PTC,c=DeveloperForZ
changeType: modify
add: uniqueMember
uniqueMember: racfID=INS010,profileType=user,cn=RACF,o=IBM,c=US
uniqueMember: racfID=INS011,profileType=user,cn=RACF,o=IBM,c=US
uniqueMember: racfID=INS012,profileType=user,cn=RACF,o=IBM,c=US
uniqueMember: racfID=INS013,profileType=user,cn=RACF,o=IBM,c=US
uniqueMember: racfID=INS014,profileType=user,cn=RACF,o=IBM,c=US
```

pushtoclient.properties

```
# BANKING and INSURANCE have different configuration needs
config.enabled=LDAP
# everyone receives product updates
product.enabled=TRUE
```



```
# only RDZADMIN can reject configuration updates
reject.config.updates=LDAP
# only RDZADMIN can reject product updates
reject.product.updates=LDAP
```

rsed.envvars

デフォルト設定が使用されるため、更新の必要はありません。

- `_RSE_LDAP_SERVER=CDFMVS08.RALEIGH.IBM.COM`
- `_RSE_LDAP_PORT=389`
- `_RSE_LDAP_PTC_GROUP_SUFFIX="o=PTC,c=DeveloperForZ"`

/var/rdz/pushtoclient/*install

グループ BANKING および INSURANCE のワークスペース構成をエクスポートする際に、エクスポート・ウィザードによって `/var/rdz/pushtoclient/grouping/<devgroup>/` ディレクトリーと、それに後続するディレクトリー構造が作成されます。

- `/var/rdz/pushtoclient/grouping/BANKING/*`
- `/var/rdz/pushtoclient/grouping/INSURANCE/*`

個別化された製品更新のシナリオは存在しないため、クライアント管理者が `/var/rdz/pushtoclient/grouping/<devgroup>のinstall/` および `install/responsefiles/` サブディレクトリーを作成または更新する必要はありません。

クライアント管理者は、製品の更新に必要な応答ファイルをデフォルト・グループ・ディレクトリー `/var/rdz/pushtoclient/install/responsefiles/` 内に作成する必要があります。

SAF ベースのグループ選択

SAF (Security Access Facility) は、あらゆる z/OS セキュリティー製品にアクセスするためのインターフェースです。Developer for System z では、このインターフェースを使用して、ご使用のセキュリティー製品を照会したり、クライアントへのプッシュに関連した情報を取得することができます。

セキュリティー・データベースの定義を選択手段として使用する場合 (`pushtoclient.properties` のディレクティブに SAF 値が指定されている場合)、Developer for System z は 表 37 にリストされているプロファイルへのアクセス権限を検証して、ユーザーが所属している開発者グループを判別したり、ユーザーが更新の拒否を許可されているかどうかを判別します。

表 37. クライアントへのプッシュの SAF 情報

FACILITY プロファイル	固定長	必要なアクセス権	結果
FEK.PTC.CONFIG.ENABLED. sysname.devgroup	23	READ	クライアントは 指定されたグル ープ用の構成の 更新を受諾する

表 37. クライアントへのプッシュの SAF 情報 (続き)

FACILITY プロファイル	固定長	必要なアクセス権	結果
FEK.PTC.PRODUCT.ENABLED. sysname.devgroup	24	READ	クライアントは指定されたグループ用の製品の更新を受諾する
FEK.PTC.REJECT.CONFIG. UPDATES.sysname	30	READ	ユーザーは構成の更新を拒否できる
FEK.PTC.REJECT.PRODUCT. UPDATES.sysname	31	READ	ユーザーは製品の更新を拒否できる

注: プロファイルへのアクセス許可がユーザーに付与されているかどうかは判断できないことをセキュリティー・ソフトウェアが示した場合、Developer for System z そのユーザーにはアクセス許可が付与されていないと仮定します。この一例として、プロファイルが定義されていない場合があります。

devgroup 値は、特定の開発者グループに割り当てられたグループ名と一致します。グループ名は、Developer for System z クライアントに表示されることに注意してください。

sysname の値は、ターゲット・システムのシステム名と一致します。

「固定長」列は、関連するセキュリティー・プロファイルの固定部分の長さについて説明しています。

デフォルトでは、Developer for System z は、FEK.* プロファイルが FACILITY セキュリティー・クラス内に存在していると想定します。FACILITY クラス内のプロファイルには 39 文字までの文字数制限があることに注意してください。プロファイルの固定部分 (FEK.PTC.<key>) の長さと、プロファイルのサイト固有部分 (sysname または sysname.devgroup) の長さの合計が、この制限数を超過する場合は、プロファイルを別のクラス内に置き、代わりにこのクラスを使用するように Developer for System z に指示してください。これを行うには、rsed.envvars にある _RSE_FEK_SAF_CLASS をコメント解除して、適切なクラス名を指定します。

サンプル・セットアップ

ある企業がシステム CDFMVS08 上で Developer for System z をアクティブにしていると仮定します。RACF セキュリティー・データベースは複数のシステム間で共有されています。このセキュリティー・データベースには以下のグループが定義されています。

- DEVBANK : 銀行アプリケーションに関する作業を行う開発者
- DEVINSUR : 保険アプリケーションに関する作業を行う開発者
- RDZADMIN : System z クライアント管理機能の開発者

各開発者グループは特定のクライアント構成ファイルを必要とします。また、すべての開発者は同一のクライアント・バージョン制御の管理下にあります。クライア

ント管理者とは異なり、開発者はクライアントへのプッシュが提示する変更を拒否することは許可されません。拒否ルールは、将来の拡張に備えて、すべてのシステムに対して有効にします。

クライアント管理者とセキュリティー管理者は、クライアントへのプッシュ・グループ名 BANKING および INSURANCE を構成の更新に使用することについて同意する必要があります。

セキュリティー定義

```
# allow RDZADMIN and DEVBANK to select push-to-client group BANKING
RDEFINE FACILITY (FEK.PTC.CONFIG.ENABLED.CDFMVS08.BANKING) -
  UACC(NONE) DATA('RATIONAL DEVELOPER FOR SYSTEM Z - PUSH-TO-CLIENT')
PERMIT FEK.PTC.CONFIG.ENABLED.CDFMVS08.BANKING CLASS(FACILITY) -
  ID(RDZADMIN DEVBANK) ACCESS(READ)

# allow RDZADMIN and DEVINSUR to select push-to-client group INSURANCE
RDEFINE FACILITY (FEK.PTC.CONFIG.ENABLED.CDFMVS08.INSURANCE) -
  UACC(NONE) DATA('RATIONAL DEVELOPER FOR SYSTEM Z - PUSH-TO-CLIENT')
PERMIT FEK.PTC.CONFIG.ENABLED.CDFMVS08.INSURANCE CLASS(FACILITY) -
  ID(RDZADMIN DEVINSUR) ACCESS(READ)

# RDZADMIN can reject configuration updates on any system
RDEFINE FACILITY (FEK.PTC.REJECT.CONFIG.UPDATES.*) -
  UACC(NONE) DATA('RATIONAL DEVELOPER FOR SYSTEM Z - PUSH-TO-CLIENT')
PERMIT FEK.PTC.REJECT.CONFIG.UPDATES.* CLASS(FACILITY) -
  ID(RDZADMIN) ACCESS(READ)

# RDZADMIN can reject product updates on any system
RDEFINE FACILITY (FEK.PTC.REJECT.PRODUCT.UPDATES.*) -
  UACC(NONE) DATA('RATIONAL DEVELOPER FOR SYSTEM Z - PUSH-TO-CLIENT')
PERMIT FEK.PTC.REJECT.CONFIG.UPDATES.* CLASS(FACILITY) -
  ID(RDZADMIN) ACCESS(READ)

# activate changes
SETROPTS RACLIST(FACILITY) REFRESH
```

pushtoclient.properties

```
# BANKING and INSURANCE have different configuration needs
config.enabled=SAF
# everyone receives product updates
product.enabled=TRUE
# only RDZADMIN can reject configuration updates
reject.config.updates=SAF
# only RDZADMIN can reject product updates
reject.product.updates=SAF
```

rsed.envvars

デフォルト設定が使用されるため、更新の必要はありません。

```
_RSE_FEK_SAF_CLASS=FACILITY
```

/var/rdz/pushtoclient/*install

グループ BANKING および INSURANCE のワークスペース構成をエクスポートする際に、エクスポート・ウィザードによって /var/rdz/pushtoclient/grouping/<devgroup>/ ディレクトリーと、それに後続するディレクトリー構造が作成されます。

- var/rdz/pushtoclient/grouping/BANKING/*
- /var/rdz/pushtoclient/grouping/INSURANCE/*

個別化された製品更新のシナリオは存在しないため、クライアント管理者が /var/rdz/pushtoclient/grouping/<devgroup>/ の install/ および install/responsefiles/ サブディレクトリーを作成または更新する必要はありません。

クライアント管理者は、製品の更新に必要な応答ファイルをデフォルト・グループ・ディレクトリー /var/rdz/pushtoclient/install/responsefiles/ 内に作成する必要があります。

変更の拒否の猶予期間

サンプル・セットアップがアクティブ状態のときに、重要な修正を含む Developer for System z 修正パッケージが使用可能になったとします。ただし、銀行プロジェクトの開発者の多くが、このタイミングで各自のワークステーションに直ちに何らかの変更を加えることに難色を示したとします。

この問題を解決するために、セキュリティー管理者は、すべての DEVBANK 開発者に対して猶予期間を与えて、開発者が更新の延期 (拒否) を選択できるようにします。

猶予期間のセットアップは、非常に単純なプロセスです。

```
# start of grace period
PERMIT FEK.PTC.REJECT.PRODUCT.UPDATES.* CLASS(FACILITY) -
  ID(DEVBANK) ACCESS(READ)
```

```
# activate changes
SETROPTS RACLIST(FACILITY) REFRESH
```

猶予期間の終了時点で、追加の権限を除去して元に戻すこともできます。

```
# end of grace period
PERMIT FEK.PTC.REJECT.PRODUCT.UPDATES.* CLASS(FACILITY) -
  ID(DEVBANK) DELETE
```

```
# activate changes
SETROPTS RACLIST(FACILITY) REFRESH
```

ホスト・ベースのプロジェクト

z/OS プロジェクトは、クライアント上の「z/OS プロジェクト」パースペクティブで個別に定義できます。または、z/OS プロジェクトをホスト上で集中的に定義してクライアントに対して個々のユーザー単位で伝搬することもできます。それらの「ホスト・ベースのプロジェクト」は、クライアント上で定義されたプロジェクトと外観も機能もまったく同じですが、クライアントは、それらの構造、メンバー、およびプロパティーを変更できず、ホストに接続している場合にのみ、それらのプロジェクトにアクセスできます。

ホスト・ベースのプロジェクトのベース・ディレクトリー (クライアント管理者が定義する) は、/var/rdz/pushtoclient/keymapping.xml で定義され、デフォルトでは /var/rdz/pushtoclient/projects になっています。

ホスト・ベースのプロジェクトを構成するには、プロジェクト・マネージャーまたは主任開発者が以下のタイプの構成ファイルを定義する必要があります。すべてのファイルは、UTF-8 でエンコードされた XML ファイルです。

- プロジェクトのインスタンス・ファイルは、単一のユーザー ID に固有であり、再使用可能なプロジェクト定義ファイルを指します。ホスト・ベースのプロジェクトを使用して作業するユーザーごとに 1 つのサブディレクトリー `/var/rdz/pushtoclient/projects/<userid>/` が必要です。これには、ダウンロードするプロジェクトごとに 1 つのプロジェクト・インスタンス・ファイル (*.hbpin) が含まれます。
- プロジェクト定義ファイルでは、プロジェクトの構造と内容を定義します。また、このファイルは複数のユーザーが再利用できます。プロジェクト定義ファイル (*.hbppd) では、そのプロジェクトに含まれるサブプロジェクトがリストされます。このファイルは、ルートのプロジェクト定義ディレクトリーか、そのディレクトリーのいずれかのサブディレクトリーに配置されます。
- サブプロジェクト定義ファイルでは、サブプロジェクトの構造と内容を定義します。また、このファイルは複数のユーザーが再利用できます。サブプロジェクト定義ファイル (*.hbpsd) では、単一のロード・モジュールを作成するために必要なリソースのセットが定義されます。このファイルは、ルートのプロジェクト定義ディレクトリーか、そのディレクトリーのいずれかのサブディレクトリーに配置されます。
- サブプロジェクト・プロパティー・ファイルは、変数置換をサポートしているプロパティー・ファイルであり、複数のサブプロジェクトで再利用できます。サブプロジェクト・プロパティー・ファイル (*.hbppr) は、変数置換をサポートしているので、複数のユーザー間でのプロパティー・ファイルの共有が可能です。このファイルは、ルートのプロジェクト定義ディレクトリーか、そのディレクトリーのいずれかのサブディレクトリーに配置されます。

ホスト・ベースのプロジェクトは、136 ページの『複数の開発者グループ』で説明されている複数グループのセットアップに加えることもできます。つまり、ホスト・ベースのプロジェクトは `/var/rdz/pushtoclient/grouping/<devgroup>/projects/` で定義することもできるということです。

ワークスペースが特定のグループにバインドされているときに、このグループとデフォルト・グループに同一ユーザーのプロジェクト定義が存在すると、そのユーザーはデフォルト・グループとその特定のグループの両方からのプロジェクト定義を受け取ります。

第 8 章 CICSTS に関する考慮事項

従来、CICS に対してリソースを定義する役割は、CICS 管理者の専門的な役割でした。アプリケーション開発者が CICS リソースを定義できるようにすることには、以下のようなさまざまな理由から抵抗がありました。

- ほとんどの CICS リソース定義には多数のパラメーターがあり、それらのパラメーターは、その複雑さ、他のリソース定義との相互関係、および作業基準のために、正しく定義するには CICS 管理者の知識が必要になります。定義が正しくないと、予期しない結果が生じる場合があります、それによって CICS 領域全体が影響を受けるおそれがあります。
- ほとんどのお客様の作業現場では、複数のアプリケーション・グループおよび開発者によって共用される必要がある CICS の開発環境とテスト環境が提供されています。多数のお客様の作業現場には、これらの環境についてのサービス・レベル・アグリーメントがあります。それらのアグリーメントを順守するには、環境を厳格に管理する必要があります。

Developer for System z では、この問題に対処するため、CICS 管理者が Application Deployment Manager の一部である CICS リソース定義 (CRD) サーバーを使用して、CICS リソース定義のデフォルトおよび CICS リソース定義パラメーターの表示プロパティを制御できるようにしています。

例えば、CICS 管理者は、アプリケーション開発者が更新できない特定の CICS リソース定義パラメーターを提供できます。その他の CICS リソース定義パラメーターは更新可能で、デフォルトが提供される場合もされない場合もあります。あるいは、無用な複雑さを避けるために、CICS リソース定義パラメーターを非表示にすることもできます。

アプリケーション開発者は、CICS リソース定義に満足できたら、それらの定義を稼働中の CICS テスト環境に直ちにインストールするか、さらに CICS 管理者による編集と承認を受けるために、マニフェストにエクスポートすることができます。CICS 管理者は、管理ユーティリティ (バッチ・ユーティリティ) またはマニフェスト処理ツールを使用して、リソース定義の変更を実装できます。

注: マニフェスト処理ツールは、IBM CICS エクスプローラー用のプラグインです。

ホスト・システムに Application Deployment Manager をセットアップするために必要となるタスクの詳細については、「[「ホスト構成ガイド」\(SC88-5663\) の「\(オプション\) Application Deployment Manager」](#)」を参照してください。

Application Deployment Manager のカスタマイズでは、以下のサービスが Developer for System z に追加されます。

- (クライアント側) IBM CICS Explorer[®]は、CICS リソースを表示および管理するための Eclipse ベースのインフラストラクチャーを提供し、CICS ツール同士のさらに緊密な統合を可能にします。
- (クライアント側) CICS リソース定義 (CRD) エディター

- (ホスト側) CICS リソース定義 (CRD) サーバー (CICS アプリケーションとして稼働)

Application Deployment Manager CICS リソース定義 (CRD) サーバーは、CRD サーバー自体と、CRD リポジトリ、関連する CICS リソース定義、および Web サービス・インターフェースを使用する場合は、Web サービス・バインド・ファイルとサンプルのパイプライン・メッセージ・ハンドラーから構成されます。CRD サーバーは、Developer for System z 資料で CICS 主接続領域として参照されている Web Owning Region (WOR) 内で稼働する必要があります。

現行リリースの Developer for System z で使用可能な Application Deployment Manager のサービスの詳細については、Developer for System z インフォメーション・センター (<http://pic.dhe.ibm.com/infocenter/ratdevz/v9r0/index.jsp>) を参照してください。

RESTful と Web サービス

CICS Transaction Server バージョン 4.1 以上では、Representational State Transfer (RESTful) の原則に従って設計された HTTP インターフェースをサポートしています。現在この RESTful インターフェースは、戦略的な CICSTS インターフェースとしてクライアント・アプリケーションで使用されています。従来の Web サービス・インターフェースはすでに安定化しており、今後は RESTful インターフェースのみが機能拡張の対象となります。

Application Deployment Manager は、この指示書に従い、Developer for System z バージョン 7.6 以上で新たに導入されたすべてのサービスに RESTful CRD サーバーを必要とします。

必要であれば、1 つの CICS 領域で RESTful インターフェースと Web サービス・インターフェースを同時にアクティブにすることができます。この場合、その領域で 2 つの CRD サーバーがアクティブになります。両サーバーは、同じ CRD リポジトリを共有します。2 番目のインターフェースを領域に対して定義すると、CICS から定義の重複に関する警告が発行されるので注意してください。

主接続領域と非主接続領域

CICS テスト環境は、いくつかの Multi-Region Option (MRO) 接続領域から構成される場合があります。時の経過と共に、それらの領域を分類するために非公式の指定が使用されてきました。一般的な指定は、Terminal Owning Region (TOR)、Web Owning Region (WOR)、Application Owning Region (AOR)、および Data Owning Region (DOR) です。

Web Owning Region は、CICS Web サービス・サポートを実装するために使用され、Application Deployment Manager CICS リソース定義 (CRD) サーバーをこの領域内で実行する必要があります。この領域は、Application Deployment Manager に対しては CICS 主接続領域として認識されます。CRD クライアントは、CICS 主接続領域への Web サービス接続を実装します。

CICS 非主接続領域は、CRD サーバーのサービス対象となる、それ以外のすべての領域です。このサービスには、IBM CICS エクスプローラーを使用したリソースの表示と、CICS リソース定義エディターを使用したリソースの定義が含まれます。

CICSplex® SM ビジネス・アプリケーション・サービス (BAS) を使用して CICS 主接続領域の CICS リソース定義を管理する場合は、BAS によって管理される、その他のすべての CICS 領域を CRD サーバーのサービス対象とすることができます。

BAS で管理されていない CICS 領域を CRD サーバーによってサービス可能にするには、追加の変更が必要です。

CICS リソース・インストール・ロギング

CRD サーバーが CICS リソースに対して行ったアクションは、CICS CSDL TD キューの中にログとして記録されます。一般に、このキューは、使用している CICS 領域の DD MSGUSR を指します。

CICSplex SM ビジネス・アプリケーション・サービス (BAS) を使用して CICS リソース定義を管理する場合、ログを作成するには、CICSplex SM EYUPARM ディレクティブ BASLOGMSG を (YES) に設定する必要があります。

Application Deployment Manager セキュリティー

CRD リポジトリ・セキュリティ

CRD サーバー・リポジトリ VSAM データ・セットは、すべてのデフォルト・リソース定義を保持しています。したがって、更新されないように保護する必要がありますが、開発者は、そこに保管された値の読み取りを許可される必要があります。CRD リポジトリを保護するためのサンプルの RACF コマンドについては、58 ページの『データ・セット・プロファイルを定義する』を参照してください。

パイプライン・セキュリティ

CICS が Web サービス・インターフェースを通じて SOAP メッセージを受信した場合、そのメッセージはパイプラインによって処理されます。パイプラインは順番に実行される一連のメッセージ・ハンドラーです。CICS は、パイプライン構成ファイルを読み取って、パイプライン内でどのメッセージ・ハンドラーを起動すればよいかを判別します。メッセージ・ハンドラーは、その中で、Web サービスの要求および応答の特別な処理を実行できるプログラムです。

Application Deployment Manager は、メッセージ・ハンドラーと SOAP ヘッダー処理プログラムの起動を指定する、サンプルのパイプライン構成ファイルを提供します。

パイプライン・メッセージ・ハンドラー (ADNTMSGH) は、SOAP ヘッダー内のユーザー ID とパスワードを処理することにより、セキュリティのために使用されます。ADNTMSGH は、サンプルのパイプライン構成ファイルによって参照されるため、CICS RPL 連結の中に入れる必要があります。

トランザクション・セキュリティ

CPIH はデフォルトのトランザクション ID で、パイプラインによって起動されたアプリケーションは、この ID の下で実行されます。多くの場合、CPIH は最小レベルの許可用に設定されます。

Developer for System z は、CICS リソースの定義および照会時に、CRD サーバーが使用する複数のトランザクションを提供します。これらのトランザクション ID は、要求された操作に応じて CRD サーバーが設定します。トランザクション ID のカスタマイズについて詳しくは、「[ホスト構成ガイド](#)」(SC88-5663) の「『(オプション) Application Deployment Manager』」を参照してください。

トランザクション	説明
ADMS	マニフェスト処理ツールからの CICS リソース変更要求用。一般に、これは CICS 管理者が使用するためのものです。このトランザクションは高いレベルの許可を必要とします。
ADMI	CICS リソースを定義、インストール、またはアンインストールする要求用。このトランザクションは、サイトのポリシーにもよりますが、中程度の許可を必要とすると考えられます。
ADMR	CICS の環境情報またはリソース情報を取り出す、上記以外のすべての要求用。このトランザクションは、サイトのポリシーにもよりますが、最小レベルの許可を必要とすると考えられます。

CRD サーバー・トランザクションによって行われるリソース定義要求の一部、または全部を、セキュリティで保護してください。最低でも、更新コマンド (デフォルトの Web サービス・パラメーター、デフォルトの記述子パラメーター、およびファイル名からデータ・セット名へのバインディング) をセキュリティで保護し、CICS 管理者のみが、グローバル・リソースのデフォルトの設定に使用されるこれらのコマンドを発行できるようにしてください。

トランザクションが接続すると、CICS リソース・セキュリティ検査機能 (使用可能な場合) により、ユーザー ID にはそのトランザクション ID を実行する許可が与えられます。

リソース検査は、稼働中のトランザクションの RESSEC オプション、RESSEC システム初期化パラメーターによって制御され、CRD サーバーの場合は XPCT システム初期化パラメーターによっても制御されます。

リソース検査は、XPCT システム初期化パラメーターの値が NO 以外で、かつ TRANSACTION 定義の RESSEC オプションが YES であるかまたは、RESSEC システム初期化パラメーターが ALWAYS である場合にのみ実行されます。

以下の RACF コマンドは、CRD サーバー・トランザクションを保護する方法の例を示しています。CICS セキュリティの定義の詳細については、「[RACF Security Guide for CICSTS](#)」を参照してください。

- RALTER GCICSTRN SYSADM UACC(NONE) ADDMEM(ADMS)
- PERMIT SYSADM CLASS(GCICSTRN) ID(#cicsadmin)
- RALTER GCICSTRN DEVELOPER UACC(NONE) ADDMEM(ADMI)
- PERMIT DEVELOPER CLASS(GCICSTRN) ID(#cicsdeveloper)
- RALTER GCICSTRN ALLUSER UACC(READ) ADDMEM(ADMR)
- SETROPTS RACLIST(TCICSTRN) REFRESH

SSL 暗号化通信

Application Deployment Manager クライアントが Web サービス・インターフェースを使用して CRD サーバーを起動するときは、データ・ストリームの SSL 暗号化がサポートされます。この通信への SSL の使用は、CICSTS TCPIPService 定義の SSL(YES) キーワードによって制御されます。詳しくは、「*RACF Security Guide for CICSTS*」を参照してください。

リソース・セキュリティ

CICSTS は、リソースを保護する機能と、リソースを操作するコマンドを提供しています。セキュリティを完全に構成していない状態でアクティブにすると、一部の Application Deployment Manager アクションが失敗する場合があります (例えば、新しいリソース・タイプを操作する権限を付与するアクションなど)。

Application Deployment Manager で機能が失敗した場合は、以下のようなメッセージがないか CICS ログを調べ、「*RACF Security Guide for CICSTS*」の説明に従って修正を行ってください。

```
DFHXS1111 %date %time %applid %tranid Security violation by user
%userid at netname %portname for resource %resource in class
%classname. SAF codes are (X'safresp',X'safreas'). ESM codes are
(X'esmpresp',X'esmpreas').
```

管理ユーティリティ

Developer for System z が提供する管理ユーティリティを使用して、CICS 管理者は CICS リソース定義のデフォルト値を指定できます。これらのデフォルトは、読み取り専用とするか、アプリケーション開発者による編集を可能にすることができます。

管理ユーティリティは、以下の機能を提供します。

- CICSplex 管理対象テスト環境の CICSplex 名
- CICSplex SM ステージング・グループ名
- マニフェスト・エクスポート規則の設定
- CICS リソース属性のデフォルトおよび表示許可

- VSAM データ・セット定義に使用される、CICS 論理から物理へのバインディング

管理ユーティリティは、データ・セット FEK.#CUST.JCL 内のサンプル・ジョブ ADNJSAPU によって呼び出されます。このユーティリティを使用するには、CRD リポジトリに対する UPDATE アクセス権が必要です。

ADNJSAPU は FEK.#CUST.JCL に置かれます。ただし、z/OS システム・プログラマーが、ジョブ FEK.SFEKSAMP(FEKSETUP)をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳しくは、「[ホスト構成ガイド](#)」(SC88-5663) の「[カスタマイズ・セットアップ](#)」を参照してください。

注: ADNJSAPU ジョブを実行する前に、CICS で CRDリポジトリをクローズする必要があります。このリポジトリは、ジョブの完了後に再びオープンすることができます。例えば、CICS にサインオンした後、以下のコマンドを入力して、ファイルをそれぞれクローズおよびオープンすることができます。

- CEMT S FILE(ADNREPF0) CLOSED
- CEMT S FILE(ADNREPF0) OPEN

CICS テスト環境の CRD リポジトリを更新するために、入力制御ステートメントが使用されます。この環境には、以下の一般的な構文規則が適用されます。

- 1 桁目のアスタリスクは、コメント行を示しています。
- DEFINE コマンドは 1 桁目から始める必要があります、直後に 1 つのスペース、その直後に有効なキーワード (TRANSACTION など) を続ける必要があります。
- キーワードの値は、キーワードの直後に続ける必要があります。間にスペースを入れることは許されません。唯一の例外は、表示許可キーワードの UPDATE、PROTECT、および HIDDEN の場合で、これらは値を持ちません。
- キーワードの値は、小括弧で囲みます。
- キーワードとその値は、1 行に収める必要があります。

以下のサンプル定義は、「[CICS Resource Definition Guide for CICSTS](#)」で定義されている DFHCSDUP コマンドの構造に従っています。唯一の相違点は、以下の表示許可キーワードが挿入されていることで、これらのキーワードは、属性値を 3 つの許可セットにグループ化するために使用されます。

UPDATE	このキーワードの直後にある属性は、Developer for System z を使用するアプリケーション開発者によって更新可能になります。これは、省略された属性のデフォルトでもあります。
PROTECT	このキーワードの直後にある属性は、表示されますが、Developer for System z を使用するアプリケーション開発者による更新から保護されます。
HIDDEN	このキーワードの直後にある属性は表示されず、Developer for System z を使用するアプリケーション開発者による更新からも保護されます。

以下の ADNJSAPU コード・サンプルを参照してください。

```
//ADNJSAPU JOB <JOB PARAMETERS>
//*
//ADNJSAPU EXEC PGM=ADNJSAPU,REGION=1M
//STEPLIB DD DISP=SHR,DSN=FEK.SFEKLOAD
//ADMREP DD DISP=OLD,DSN=FEK.#CUST.ADNREPFO
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
*
* CICSplex SM parameters
*
DEFINE CPSMNAME( )
*DEFINE STAGINGGROUPNAME(ADMSTAGE)
*
* Manifest export rule
*
DEFINE MANIFESTEXPORTRULE(installOnly)
*
* CICS resource definition defaults
* Omitted attributes default to UPDATE.
*
* DB2TRAN default attributes
*
DEFINE DB2TRAN()
    UPDATE DESCRIPTION()
        ENTRY()
        TRANSID()
*
* DOCTEMPLATE default attributes
*
DEFINE DOCTEMPLATE()
    UPDATE DESCRIPTION()
        TEMPLATENAME()
        FILE() TSQUEUE() TDQUEUE() PROGRAM() EXITPGM()
        DDNAME(DFHHTML) MEMBERNAME()
        HFSFILE()
        APPENDCRLF(YES) TYPE(EBCDIC)
*
* File default attributes
*
DEFINE FILE()
    UPDATE DESCRIPTION()
        RECORDSIZE() KEYLENGTH()
        RECORDFORMAT(V) ADD(NO)
        BROWSE(NO) DELETE(NO) READ(YES) UPDATE(NO)
        REMOTESYSTEM() REMOTENAME()
    PROTECT DSNAM() RLSACCESS(NO) LSRPOOLID(1) STRINGS(1)
        STATUS(ENABLED) OPENTIME(FIRSTREF)
        DISPOSITION(SHARE) DATABUFFERS(2) INDEXBUFFERS(1)
        TABLE(NO) MAXNUMRECS(NOLIMIT)
        READINTEG(UNCOMMITTED) DSNSHARING(ALLREQS)
        UPDITEMODEL(LOCKING) LOAD(NO)
        JNLREAD(NONE) JOURNAL(NO)
        JNLSYNCREAD(NO) JNLUPDATE(NO)
        JNLADD(NONE) JNLSYNCSWRITE(YES)
        RECOVERY(NONE) FWDRECOVLOG(NO)
        BACKUPTYPE(STATIC)
        PASSWORD() NSRGROUP()
        CFDTPOOL() TABLNAME()
```

図 27. ADNJSAPU - CICS 管理ユーティリティー

```

*
* Mapset default attributes
*
DEFINE MAPSET()
    UPDATE DESCRIPTION()
    PROTECT RESIDENT(NO) STATUS(ENABLED)
        USAGE(NORMAL) USELPACOPY(NO)
** Processtype default attributes
*
DEFINE PROCESSTYPE()
    UPDATE DESCRIPTION()
        FILE(BTS)
    PROTECT STATUS(ENABLED)
        AUDITLOG() AUDITLEVEL(OFF)
*
* Program default attributes
*
DEFINE PROGRAM()
    UPDATE DESCRIPTION()
        CEDF(YES) LANGUAGE(LE370)
        REMOTESYSTEM() REMOTENAME() TRANSID()
    PROTECT API(CICSAPI) CONCURRENCY(QUASIRENT)
        DATALOCATION(ANY) DYNAMIC(NO)
        EXECKEY(USER) EXECUTIONSET(FULLAPI)
        RELOAD(NO) RESIDENT(NO)
        STATUS(ENABLED) USAGE(NORMAL) USELPACOPY(NO)
        HIDDEN JVM(NO) JVMCLASS() JVMPROFILE(DFHJVMPR)
*
* TDQueue default attributes
*
DEFINE TDQUEUE()
    UPDATE DESCRIPTION()
        TYPE(INTRA)
* Extra partition parameters
    DDNAME() DSNAME()
        REMOTENAME() REMOTESYSTEM() REMOTELength(1)
        RECORDSIZE() BLOCKSIZE(0) RECORDFORMAT(UNDEFINED)
        BLOCKFORMAT() PRINTCONTROL() DISPOSITION(SHR)
* Intra partition parameters
    FACILITYID() TRANSID() TRIGERRLEVEL(1)
    USERID()
* Indirect parameters
    INDIRECTNAME()
    PROTECT WAIT(YES) WAITACTION(REJECT)
* Extra partition parameters
    DATABUFFERS(1)
    SYSOUTCLASS() ERROROPTION(IGNORE)
    OPENTIME(INITIAL) REWIND(LEAVE) TYPEFILE(INPUT)
* Intra partition parameters
    ATIFACILITY(TERMINAL) RECOVSTATUS(NO)

```

図 28. ADNJSAPU - CICSTS 管理ユーティリティ (2/3)

```

*
* Transaction default attributes
*
DEFINE TRANSACTION()
  UPDATE  DESCRIPTION()
          PROGRAM()
          TWASIZE(0)
          REMOTESYSTEM() REMOTENAME() LOCALQ(NO)
  PROTECT PARTITIONSET() PROFILE(DFHCICST)
          DYNAMIC(NO) ROUTABLE(NO)
          ISOLATE(YES) STATUS(ENABLED)
          RUNAWAY(SYSTEM) STORAGECLEAR(NO)
          SHUTDOWN(DISABLED)
          TASKDATAKEY(USER) TASKDATALOC(ANY)
          BREXIT() PRIORITY(1) TRANCLASS(DFHTCL00)
          DTIMOUT(NO) RESTART(NO) SPURGE(NO) TPURGE(NO)
          DUMP(YES) TRACE(YES) CONFDATA(NO)
          OTSTIMEOUT(NO) WAIT(YES) WAITTIME(00,00,00)
          ACTION(BACKOUT) INDOUBT(BACKOUT)
          RESSEC(NO) CMDSEC(NO)
          TRPROF()
          ALIAS() TASKREQ()
          XTRANID() TPNAME() XTPNAME()

*
* URDIMAP attributes
*
DEFINE URIMAP()
  UPDATE  USAGE(CLIENT)
          DESCRIPTION()
          PATH(/required/path)
          TCPIPSERVICE()
          TRANSACTION()
          PROGRAM()
  PROTECT ANALYZER(NOANALYZER)
          ATOMSERVICE()
          CERTIFICATE()
          CHARACTERSET()
          CIPHERS()
          CONVERTER()
          HFSFILE()
          HOST(host.mycompany.com)
          HOSTCODEPAGE()
          LOCATION()
          MEDIATYPE()
          PIPELINE()
          PORT(NO)
          REDIRECTTYPE(NONE)
          SCHEME(HTTP)
          STATUS(ENABLED)
          TEMPLATENAME()
          USERID()
          WEBSERVICE()

*
* Optional file name to VSAM data set name binding
*
*DEFINE DSBINDING() DSNAME()
/*

```

図 29. ADNJSAPU - CICSTS 管理ユーティリティー (3/3)

管理ユーティリティーのマイグレーションに関する注

Developer for System z バージョン 7.6.1 では、管理ユーティリティーに URIMAP サポートが追加されています。URIMAP サポートを使用できるようにするには、

CRD リポジトリ VSAM データ・セットを最大レコード・サイズ 3000 で割り振る必要があります。Developer for System z バージョン 7.6.1 までは、サンプルの CRD リポジトリ割り振りジョブは、最大レコード・サイズ 2000 を使用します。

以前の CRD リポジトリを使用している場合に URIMAP サポートを使用可能にするには、以下のステップを実行します。

1. 既存の CRD リポジトリ FEK.#CUST.ADNREPF0 のバックアップを作成します。
2. 既存の CRD リポジトリを削除します。
3. 新しい CRD リポジトリの割り振りと初期化を行うために、ジョブ FEK.SFEKSAMP(ADNVCRD) をカスタマイズして実行依頼します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。
4. 管理ユーティリティを使用して新しい CRD リポジトリにデータを取り込むために、ジョブ FEK.SFEKSAMP(ADNJSPAU) をカスタマイズして実行依頼します。

注:

- 既存の CRD リポジトリをマイグレーションする必要はありません。これは、管理ユーティリティが、実行されるたびに CRD リポジトリの内容全体を置換するからです。
- CRD リポジトリにはバージョンの互換性の問題がありません。サポートされる Developer for System z クライアント・コードおよびホスト・コードはすべて、どちらの最大レコード・サイズにも対応します。ただし、最大レコード・サイズが 3000 ではない場合は、URIMAP サポートが使用不可になります。

管理ユーティリティのメッセージ

以下のメッセージは、管理ユーティリティが SYSPRINT DD に対して発行します。メッセージ CRAZ1803E、CRAZ1891E、CRAZ1892E、および CRAZ1893E は、ファイル状況コード、VSAM 戻りコード、VSAM 機能コード、および VSAM フィードバック・コードを含んでいます。VSAM 戻りコード、機能コード、およびフィードバック・コードについては、「*DFSMS Macro Instructions for Data Sets*」(SC26-7408) に説明があります。ファイル状況コードについては、「*Enterprise COBOL for z/OS 言語解説書*」(SC88-9117) に説明があります。

CRAZ1800I

行 <最後の制御ステートメントの行番号> で正常に完了しました。

(completed successfully on line <last control statement line number>)

説明: システム・プログラマー管理ユーティリティは、正常に完了しました。

ユーザー応答: なし。

CRAZ1801W

行 <最後の制御ステートメントの行番号> で警告によって完了しました。

(completed with warnings on line <last control statement line number>)

説明: システム・プログラマー管理ユーティリティは、制御ステートメントの処理時に 1 つ以上の警告を検出して完了しました。

ユーザー応答: 他の警告メッセージを調べてください。

CRAZ1802E

行 <行番号> でエラーを検出しました。(encountered an error on line <line number>)

説明: システム・プログラマー管理ユーティリティは、重大エラーを検出しました。

ユーザー応答: 他の警告メッセージを調べてください。

CRAZ1803E

リポジトリ・オープン・エラー、状況=<ファイル状況コード>
RC=<VSAM 戻りコード> FC=<VSAM 機能コード> FB=<VSAM フィード
バック・コード> (Repository open error, status=<file status code>
RC=<VSAM return code> FC=<VSAM function code> FB=<VSAM
feedback code>)

説明: システム・プログラマー管理ユーティリティは、CRD リポジトリ
をオープンしようとして重大エラーを検出しました。

ユーザー応答: VSAM 状況コード、戻りコード、機能コード、およびフィ
ードバック・コードを調べてください。

CRAZ1804E

行 <行番号> の入力レコードを認識できませんでした。(Unrecognized input
record on line <line number>)

説明: システム・プログラマー管理ユーティリティは、認識できない入力
制御ステートメントを検出しました。

ユーザー応答: DEFINE コマンドの直後に 1 つのスペースと、その直後に
CPSMNAME、STAGINGGROUPNAME、MANIFESTEXPORTRULE、DSBINDING、
DB2TRAN、DOCTEMPLATE、FILE、MAPSET、PROCESSTYPE、PROGRAM、TDQUEUE、
TRANSACTION のいずれかのキーワードがあったかどうかを調べてください。

CRAZ1805E

行 <行番号> のキーワード <キーワード> を処理しています。(Processing
keyword <keyword> on line <line number>)

説明: システム・プログラマー管理ユーティリティは、DEFINE キーワ
ード入力制御ステートメントを処理しています。

ユーザー応答: なし。

CRAZ1806E

行 <行番号> のマニフェスト・エクスポート規則が無効です。(Invalid
manifest export rule on line <line number>)

説明: システム・プログラマー管理ユーティリティは、無効なマニフェス
ト・エクスポート規則を検出しました。

ユーザー応答: MANIFESTEXPORTRULE キーワードの値が
「installOnly」、「exportOnly」、「both」のいずれかであることを確認して
ください。

CRAZ1807E

行 <行番号> に DSNAME キーワードがありません。(Missing DSNAME
keyword on line <line number>)

説明: システム・プログラマー管理ユーティリティーが処理しようとした DEFINE DSBINDING 制御ステートメントに DSNAME キーワードがありませんでした。

ユーザー応答: DEFINE DSBINDING 制御ステートメントに DSNAME キーワードが含まれているかどうか調べてください。

CRAZ1808E

行 <行番号> のキーワード <キーワード> のキーワード値が無効です。
(Invalid keyword value for keyword <keyword> on line <line number>)

説明: システム・プログラマー管理ユーティリティーが DEFINE 制御ステートメントを処理しようとして、指定されたキーワードに無効な値を検出しました。

ユーザー応答: 指定されたキーワードの長さや値が正しいかどうか調べてください。

CRAZ1890W

行 <行番号> でのキーワード構文エラー。(Keyword syntax error on line <line number>)

説明: システム・プログラマー管理ユーティリティーが DEFINE 制御ステートメントを処理しようとして、キーワードまたはキーワード値の構文エラーを検出しました。

ユーザー応答: キーワード値が小括弧で囲まれていて、キーワードの直後に存在することを確認してください。キーワードおよびキーワード値は両方とも同じ行になければなりません。

CRAZ1891W

リポジトリ重複キー書き込みエラー、状況=<ファイル状況コード>
RC=<VSAM 戻りコード> FC=<VSAM 機能コード> FB=<VSAM フィードバック・コード>。(Repository duplicate key write error, status=<file status code> RC=<VSAM return code> FC=<VSAM function code> FB=<VSAM feedback code>)

説明: システム・プログラマー管理ユーティリティーは、CRD リポジトリに書き込もうとして重複キー・エラーを検出しました。

ユーザー応答: VSAM 状況コード、戻りコード、機能コード、およびフィードバック・コードを調べてください。

CRAZ1892W

リポジトリ書き込みエラー、状況=<ファイル状況コード> RC=<VSAM 戻りコード> FC=<VSAM 機能コード> FB=<VSAM フィードバック・コード>。(Repository write error, status=<file status code> RC=<VSAM return code> FC=<VSAM function code> FB=<VSAM feedback code>)

説明: システム・プログラマー管理ユーティリティーは、CRD リポジトリに書き込もうとして重大エラーを検出しました。

ユーザー応答: VSAM 状況コード、戻りコード、機能コード、およびフィードバック・コードを調べてください。

CRAZ1893W

リポジトリ読み取りエラー、状況=<ファイル状況コード> RC=<VSAM

戻りコード> FC=<VSAM 機能コード> FB=<VSAM フィードバック・コード>。(Repository read error, status=<file status code> RC=<VSAM return code> FC=<VSAM function code> FB=<VSAM feedback code>)

説明: システム・プログラマー管理ユーティリティーは、CRD リポジトリから読み取ろうとして重大エラーを検出しました。

ユーザー応答: VSAM 状況コード、戻りコード、機能コード、およびフィードバック・コードを調べてください。

CICS トランザクションのデバッグ

CICS トランザクションをデバッグするには、統合デバッガーで以下のように CICS の更新が必要です。

- CICS JCL の更新:

- FEK,SFEKAUTH ロード・ライブラリーが LINKLIST 内がない場合、このライブラリーを領域の DFHRPL DD ステートメントで定義します。
- SYS1.SIEAMIGE ロード・ライブラリーが LINKLIST 内がない場合、このライブラリーを領域の STEPLIB DD ステートメントで定義します。

- CICS CSD の更新:

AQECSD サンプル CSD アップデート・ジョブの記述に従って、デバッガーを CICS 領域に定義します。AQECSD は FEK.#CUST.JCL に置かれます。ただし、z/OS システム・プログラマーが、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳しくは、「ホスト構成ガイド」(SC88-5663) の『カスタマイズのセットアップ』を参照してください。

読み取り専用メモリーにロードされた CICS トランザクションをデバッグするには、統合デバッガーで以下のシステム更新が必要です。

- 統合デバッガー監視プログラム呼び出し (SVC) をシステムに定義します。詳しくは、「ホスト構成ガイド」(SC88-5663) の『PARMLIB の変更』を参照してください。
- SVC を問題プログラム状態 (不許可) 環境で使用する場合、ユーザーはセキュリティー・プロファイルへの許可を必要とします。詳しくは、42 ページの『デバッグ・セキュリティー』を参照してください。

指定された CICS 領域でアクティブにできる言語環境プログラム (Language Environment (LE)) ベースのデバッガーは 1 つのみであることに注意してください。LE ベースのデバッガーを明確に示すのは、CEEEVDBG ロード・モジュールまたは別名を提供し、それがアプリケーションで使用可能でなければならないことです。

第 9 章 ユーザー出口に関する考慮事項

この章は、出口ルーチンの作成による Developer for System z の機能強化についてユーザーを支援します。

Developer for System z は、選択された Developer for System z イベントに対して出口点を提供します。出口点は、1 つの機能の処理での特定点であり、ここでその機能が出口ルーチン（ある場合）を呼び出します。出口ルーチンを作成することで追加の処理を実行できます。

従来の多くの出口点とは異なり、Developer for System z 出口点は、機能の動作の変更ができないことに注意してください。出口ルーチンが存在する場合、その出口ルーチンは機能が完了すると非同期的に呼び出されます。Developer for System z 処理は出口ルーチンの終了を待ちません。また、完了状況もチェックしません。

ユーザー出口の特性

ユーザー出口の活動化

ユーザー出口は、rsed.envvars の `_RSE_JAVA_OPTS <exit_point>.action` 変数でアクティブ化されます。ここで、`<exit_point>` は、168 ページの『使用可能な出口点』に説明されているように、特定の出口点を識別するキーワードを表しています。

```
#_RSE_JAVA_OPTS="$_RSE_JAVA_OPTS -D<exit_point>.action=<user_exit>"
```

デフォルトでは、すべての出口点は使用不可になっています。ユーザー出口ルーチンの絶対パス名をアンコメントして、出口点を使用可能にしてください。

```
#_RSE_JAVA_OPTS="$_RSE_JAVA_OPTS -D<exit_point>.action.id=<userid>"
```

デフォルトでは、RSE デーモンの割り当てられたユーザー ID は指定された出口ルーチンを実行するために使用されます。ユーザー ID をアンコメントして指定し、ユーザー出口を実行するためにその指定した ID を使用してください。パスワードを指定する必要はありません。RSE は、指定されたユーザー ID に切り替えるときにパスワードとして使用するパスチケットを生成します。

ユーザー出口ルーチンの作成

ユーザー出口ルーチンは、場合により 1 つ以上の引数が付いた z/OS UNIX シェル・コマンドとして呼び出されます。これは、開発する出口ルーチンが z/OS UNIX コマンド行から実行可能でなければならないことを意味します。共通コーディング技法には、z/OS UNIX シェル・スクリプトおよび z/OS UNIX REXX exec が含まれますが、C/C++ のようなコンパイルされたコードも使用可能です。

z/OS UNIX シェル・スクリプトについて詳しくは、「UNIX System Services ユーザーズ・ガイド」(SA88-8640)を参照してください。REXX 言語に対する z/OS

UNIX 固有の拡張子について詳しくは、「REXX および z/OS UNIX システム・サービスの使い方」(SA88-8644)を参照してください。

出口ルーチンは特殊の権限 (パスチケットを生成することができる RSE 開始タスクのユーザー ID など) をもつユーザー ID によって実行することが可能です。したがって、不正使用を防止するために出口ルーチンに対する更新権限を制限することは重要です。次のサンプルの z/OS UNIX コマンドは、書き込み権限を所有者のみに制限します。ただし、スクリプトの読み取りと実行は誰でも可能です。

```
$ chmod 755 process_logon.sh
$ ls -l process_logon.sh
-rwxr-xr-x 1 IBMUSER SYS1          2228 Feb 28 23:44 process_logon.sh
```

rsed.envvars の定義は、環境変数としてユーザー出口ルーチンに使用可能です。

RSE は、単一の引数ストリングを持つユーザー出口ルーチンを呼び出します。引数ストリングは、複数のブランクで区切られたキーワードと値をもつ単一値でも単一のストリングでもかまいません。詳しくは、168 ページの『使用可能な出口点』を参照してください。

コンソール・メッセージ

Developer for System z はコンソール・メッセージ ID FEK910I を使用して、ユーザー出口に関連するデータを表示します。

出口ルーチンの呼び出しは、次のコンソール・メッセージでマークされます。

```
FEK910I <EXIT_POINT> EXIT: invoking <exit_point> processing exit
      in thread <thread_id>
```

stdout (シェル・スクリプトの **echo** コマンド、REXX exec の **say** コマンド) に書き込まれたすべてのデータは次のコンソールに送られます。

```
FEK910I <EXIT_POINT> EXIT: <message>
```

出口ルーチンの出口は、次のコンソール・メッセージでマークされます。

```
FEK910I <EXIT_POINT> EXIT: completed <exit_point> processing exit
      in thread <thread_id>
```

可変ユーザー ID を使用した実行

Developer for System z では、開始タスクのユーザー ID または指定されたユーザー ID を使用して、出口ルーチンを実行できます。ただし、ログオン出口ルーチンのクライアント・ユーザーなどの別のユーザー ID を使用する出口ルーチンでは、なんらかのアクションの実行が必要となる場合があります。これは、以下のサンプルに示されているように、標準の z/OS UNIX サービスを使用して実行できます。

z/OS UNIX シェル・スクリプト

「UNIX System Services コマンド解説書」(SA88-8641)で説明されているように、z/OS UNIX ではスーパーユーザーまたは別のユーザーの特権を使用するための **su** コマンドが提供されています。**su** コマンドを使用する際の留意点はほとんどありません。

- **su** コマンドを実行するユーザー ID には、<userid> によって識別されたユーザー ID にパスワードを指定せずに切り替えることができるように、ご使用のセキ

セキュリティ製品の SURROGAT クラス内に BPX.SRV.<userid> プロファイルに対しての読み取り権限が備わっていなければなりません。

- **su** コマンドは新しいシェルを開始します。したがって、シェル・スクリプト内の残りのコマンドは、**su** コマンド出口によってシェルが開始されるまで実行されません。**su** コマンドによって開始された新しいシェルでステージ・コマンドを実行するために、以下の例のように、**echo** コマンドを使用して必要なコマンドおよびパイプ・コマンド文字を作成し、新しいシェルに送ることができます。特殊文字のエスケープ処理には標準のシェル・スクリプト規則が適用されることに注意してください。

```
#!/bin/sh
myID=ibmuser
echo a $(id)
echo 'echo b $(id)' | su -s $myID
echo "echo c \"$(id)\" | su -s $myID
cat /u/ibmuser/iefbr14
echo "submit /u/ibmuser/iefbr14" | su -s $myID
```

開始タスクユーザー ID により実行されるこのサンプルのログオン出口プログラムは、以下のコンソール・メッセージを生成します。

```
+FEK910I LOGON EXIT: invoking logon processing exit in thread 411
+FEK910I LOGON EXIT: a uid=8(STCRSE) gid=1(STCGRP)
+FEK910I LOGON EXIT: b uid=1(IBMUSER) gid=0(SYS1)
+FEK910I LOGON EXIT: c uid=1(IBMUSER) gid=0(SYS1)
+FEK910I LOGON EXIT: //IEFBR14 JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
+FEK910I LOGON EXIT: //IEFBR14 EXEC PGM=IEFBR14
$HASP100 IEFBR14 ON INTRDR FROM STC03919
IBMUSER
IRR010I USERID IBMUSER IS ASSIGNED TO THIS JOB.
+FEK910I LOGON EXIT: JOB JOB03926 submitted from path '/u/ibmuser/iefbr14'
ICH70001I IBMUSER LAST ACCESS AT 00:46:13 ON MONDAY, MARCH 19, 2012
$HASP373 IEFBR14 STARTED - INIT 2 - CLASS A - SYS CD08
IEF403I IEFBR14 - STARTED - TIME=00.46.14
+FEK910I LOGON EXIT: completed logon processing exit in thread 411
IEFBR14 IEFBR14 IEFBR14 0000
IEF404I IEFBR14 - ENDED - TIME=00.46.14
$HASP395 IEFBR14 ENDED
$HASP309 INIT 2 INACTIVE ***** C=BA
```

z/OS UNIX REXX exec

「REXX および *and z/OS UNIX System Services の使い方*」(SA88-8644) で説明されているように、z/OS UNIX には、現行プロセスの有効な UID を設定する **seteuid** SYSCALL コマンドが用意されています。**seteuid** コマンドを使用する際の留意点はほとんどありません。

- **seteuid** コマンドは、MVS ユーザー ID でなく、z/OS UNIX UID を使用します。最初にターゲット・ユーザー ID の UID を決める必要があります。これは **getpwnam** SYSCALL コマンドを使用して実行できます。
- **seteuid** コマンドを実行するユーザー ID には、<userid> によって識別されたユーザー ID にパスワードを指定せずに切り替えることができるように、ご使用のセキュリティ製品の SURROGAT クラス内に BPX.SRV.<userid> プロファイルに対しての読み取り権限が備わっていなければなりません。複数のユーザー ID が同じ UID を共有する場合、どのユーザー ID が検査されるかを判別する方法がないことに注意してください。

```

/* rexx */
myID='ibmuser'
say userid()
address SYSCALL 'getpwnam' myID 'pw.'
say pw.1 pw.2 pw.3 pw.4 pw.5
address SYSCALL 'seteuid' pw.2 /* PW_UID = 2 */
say retval errno errnojr
say userid()

```

開始タスクユーザー ID により実行されるこのサンプルのログオン出口プログラムは、以下のコンソール・メッセージを生成します。

```

+FEK910I LOGON EXIT: invoking logon processing exit in thread 515
+FEK910I LOGON EXIT: STCRSE
+FEK910I LOGON EXIT: IBMUSER 1 0 / /bin/sh
+FEK910I LOGON EXIT: 0 0 0
+FEK910I LOGON EXIT: IBMUSER
+FEK910I LOGON EXIT: completed logon processing exit in thread 515

```

使用可能な出口点

以下の出口点が Developer for System z で提供されています。

- 『audit.action』
- 『logon.action』

audit.action

- タイミング:

監査ユーザー出口は、アクティブ監査ログ・ファイルがクローズされるときに呼び出されます。(監査は、新しい監査ログ・ファイルに切り替えられた RSE として続行します。)

- 呼び出し引数 (1):

– <audit_log>: クローズした監査ログ・ファイルの絶対パス名

- サンプル:

/usr/lpp/rdz/samples/process_audit.rex

このサンプルの z/OS UNIX REXX exec は、クローズされた監査ログを処理するバッチ・ジョブをビルドします。

logon.action

- タイミング:

ログオン・ユーザー出口は、ユーザーがログオン・プロセスを完了したときに呼び出されます。

- 呼び出し引数 (6):

– -i <userid>: クライアントのユーザー ID。大/小文字はクライアントが入力するとおりです。

– -u <user_log_path>: このクライアントのユーザー・ログが保持されるディレクトリー

– -s <server_log_path>: サーバー・ログが保持されるディレクトリー

- -c <config_path>: 構成ファイルが保持されるディレクトリー
- -b <binaries_path>: Developer for System z がインストールされたディレクトリー
- -p <port>: RSE デーモン・ポート

- サンプル:

/usr/lpp/rdz/samples/process_logon.sh

このサンプルの z/OS UNIX シェル・スクリプトはログオン・メッセージをコンソールへ書き込みます。

第 10 章 カスタマイズ、TSO 環境の

この章は、Developer for System z で TSO 環境に DD ステートメントとデータ・セットを追加することにより、TSO ログオン・プロシージャーを模倣するのに役立ちます。

TSO コマンド・サービス

TSO コマンド・サービスは、TSO コマンドと (バッチ) ISPF コマンドを実行し、結果を要求側クライアントへ返す Developer for System z コンポーネントです。これらのコマンドは、本製品が非明示的に要求するか、ユーザーが明示的に要求できます。

Developer for System z で提供されるサンプル・メンバーは、最小の TSO/ISPF 環境を作成します。ご使用のワークショップ内の開発者がカスタム・ライブラリーまたはサード・パーティー・ライブラリーへのアクセスを必要とする場合、z/OS システム・プログラマーは必要な DD ステートメントおよびライブラリーを TSO コマンド・サービス環境に追加する必要があります。Developer for System z での実装は異なりますが、その背後にあるロジックは TSO ログオン・プロシージャーと同一です。

注: TSO コマンド・サービスは非対話式のコマンド行ツールです。このため、データの入力を要求したり ISPF パネルを表示したりするコマンドまたはプロシージャーは、機能しません。これらを実行するためには、3270 エミュレーター (Developer for System z クライアントの一部である Host Connect Emulator など) が必要です。

アクセス方式

バージョン 7.1 以降の Developer for System z では、TSO コマンド・サービスへのアクセス方法を選択できます。

- ISPF の TSO/ISPF クライアント・ゲートウェイ・サービス。これは最小の ISPF サービス・レベルを必要とします。これが、提供されたサンプルで使用されるデフォルトの方式です。
- APPC トランザクション (バージョン 7.1 より前のリリースと同様)。この方式は推奨されません。

注:

- ISPF の TSO/ISPF クライアント・ゲートウェイ・サービスは、バージョン 7.1 で使用されていた SCLM Developer Toolkit 機能の代わりになります。
- Developer for System z による APPC の使用は非推奨のマークが付いています。APPC 関連の情報は、本書では削除されています。詳しくは、ホワイト・ペーパー「*Using APPC to provide TSO command services (SC14-7291)*」を参照してください。これは、Developer for System z ライブラリー <http://www-01.ibm.com/support/docview.wss?uid=swg27038517> で入手可能です。

バージョン 7.1 以上のホストに、どのアクセス方式が使用されているかを判別するには、`rsed.envvars` を調べます。構成プロセスのときにデフォルトが使用された場合、`rsed.envvars` は `/etc/rdz/` にあります。

- `_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DTSO_SERVER=APPC"` ステートメントが存在しない (またはコメント化されたままである) 場合は、ISPF の TSO/ISPF クライアント・ゲートウェイ・サービスが使用されます。
- `_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DTSO_SERVER=APPC"` ステートメントが存在する (しかも、コメント化されていない) 場合は、APPC が使用されます。

TSO/ISPF クライアント・ゲートウェイ・アクセス方式の使用

ISPF.conf

ISPF.conf 構成ファイル (デフォルトでは `/etc/rdz/` に置かれます) は、Developer for System z に使用する TSO/ISPF 環境を定義します。アクティブな ISPF.conf 構成ファイルは 1 つだけ存在し、すべての Developer for System z ユーザーによって使用されます。

この構成ファイルのメイン・セクションでは、次のサンプルに示すように、DD 名および関連するデータ・セット連結が定義されています。

```
sysproc=ISP.SISPCLIB,FEK.SFEKPROC
ispmllib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
ispllib=ISP.SISPLOAD
myDD=HLQ1.LLQ1,HLQ2.LLQ2
```

- 1 つの DD 定義は正確に 1 行を使用し (複数行はサポートされていません)、行の長さに制限はありません。
- 定義に大/小文字の区別はなく、空白文字はすべて無視されます。
- コメント行はアスタリスク (*) で始まります。
- DD 名の直後には等号 (=) があり、その直後にデータ・セット連結が続きます。複数のデータ・セット名は、コンマ (,) によって分離されます。
- データ・セット連結は、リストされた順序で検索されます。
- データ・セットは完全修飾でなければならない、引用符 (') で囲まれていたり変数を使用されていたりしてはなりません。
- すべてのデータ・セットは、`DISP=SHR` によって割り振られます。
- 新しい DD 名は、任意に追加できますが、DD 名の (JCL) 規則に従っている必要があり、ISPF.conf 内にある別の構成パラメーターと競合してはなりません。また、TSO/ISPF クライアント・ゲートウェイ・サービスによって、ISPPROF が動的に割り振られます (`DISP=NEW,DELETE`)。

既存の ISPF プロファイルの使用

デフォルトでは、TSO/ISPF クライアント・ゲートウェイは TSO コマンド・サービス用に一時的な ISPF プロファイルを作成します。しかし、既存の ISPF プロファ

イルのコピーを使用するように、TSO/ISPF クライアント・ゲートウェイ z に指示することができます。ここで重要なものは、rsed.envvars 内の _RSE_ISPF_OPTS ステートメントです。

```
#_RSE_ISPF_OPTS="$_RSE_ISPF_OPTS&ISPPROF=&SYSUID..ISPPROF"
```

この機能を使用するには、このステートメントをコメント解除 (先行ポンド記号 (#) 文字を除去) し、既存の ISPF プロファイルの完全修飾データ・セット名を指定します。

データ・セット名の中で、以下の変数を使用できます。

- &SYSUID。開発者のユーザー ID の代わりに使用します。
- &SYSPREF。開発者の TSO 接頭部の代わりに使用します。
- &SYSNAME。IEASYMxx parmlib メンバー内で指定されているように、システム名の代わりに使用します。

注:

- "ISPPROF" で渡されたデータ・セット名が無効な場合は、一時的な空の ISPF プロファイルが代わりに使用されます。
- ISPF プロファイル (一時プロファイルとコピーされたプロファイルの両方) は、セッションの終了時に削除されます。プロファイルに加えられた変更は、既存の ISPF プロファイルにマージされません。

使用、割り振り exec の

ISPF.conf 内の allocjob ステートメント (これは、デフォルトではコメント化されています) は、ある exec を指しており、この exec を使用すると、ユーザー ID 別に詳細なデータ・セット割り振りを指定できます。

```
*allocjob = ISP.SISPSAMP(ISPZISP2)
```

この機能を使用するには、このステートメントをコメント解除 (先行アスタリスク (*) 文字を除去) し、割り振り exec への完全修飾参照を指定してください。

- この exec は ISPPROF、および ISPF.conf 内で定義された DD の割り振りの後で、ただし ISPF が初期化される前に、実行されます。割り振り exec によってこれらの定義が取り消されないようにしてください。
- この exec には 1 つのパラメーター、つまり呼び出し元のユーザー ID が渡されます。
- サンプル exec の CRAISPRX がサンプル・ライブラリー FEK.#CUST.CNTL に入っています。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳しくは、「ホスト構成ガイド」(SC88-5663) の"『カスタマイズ・セットアップ』"を参照してください。

注: この exec は ISPF が初期化される前に呼び出されるので、VPUT および VGET は使用できません。しかし、PDS(E) または VSAM ファイルを使用して、これらの機能の独自の実装を作成できます。

複数の割り振り exec の使用

ISPF.conf は 1 つの割り振り exec の呼び出しだけをサポートしていますが、その exec が別の exec を呼び出すことに制限はありません。また、パラメーターとして渡されるクライアントのユーザー ID によって、個人別の割り振り exec を呼び出すことができます。例えば、メンバー USERID'.EXEC(ALLOC) ' が存在するかどうかを検査し、実行することができます。

この機能を上手に利用すれば、次のようにして既存の TSO ログオン・プロシージャーを使用できます。

- ユーザー固有の構成ファイル (USERID'.FEKPROF' など) を読み取ります。
- どのログオン・プロシージャーがファイル内に記述されているかを調べます。
- 記述されているプロシージャーを SYS1.PROCLIB から読み取り、構文解析して、中の DD ステートメントとデータ・セット割り振りを見つけます。
- データ・セットを実際のログオン・プロシージャーと同様な方法で割り振ります。

複数の Developer for System z セットアップでの複数の ISPF.conf ファイル

前記のセクションで説明した割り振り exec シナリオで特定の要求を処理できない場合は、Developer for System z の RSE 通信サーバーのさまざまなインスタンスを作成し、それぞれに独自の ISPF.conf ファイルを使用することができます。以下に述べる方式の主な欠点は、Developer for System z ユーザーが、希望の TSO 環境を得るために同じホスト上の異なるサーバーに接続する必要があることです。

注: RSE サーバーの 2 番目のインスタンスを作成するために必要なことは、各構成ファイル、始動 JCL、および開始タスク定義を複製して更新することだけです。製品の新規インストールは必要なく、コードを複製する必要もありません。

```
$ cd /etc/rdz
$ mkdir /etc/rdz/tso2
$ cp rsed.envvars /etc/rdz/tso2
$ cp ISPF.conf /etc/rdz/tso2
$ ls /etc/rdz/tso2
ISPF.conf          rsed.envvars
$ oedit /etc/rdz/tso2/rsed.envvars
-> change: _RSE_RSED_PORT=4037
-> change: CGI_ISPCONF=/etc/rdz/tso2
-> change: -Ddaemon.log=/var/rdz/logs/tso2
-> change: -Duser.log=/var/rdz/logs/tso2
-> add at the END:
# -- NEEDED TO FIND THE REMAINING CONFIGURATION FILES
CFG_BASE=/etc/rdz
CLASSPATH=.:$CFG_BASE:$CLASSPATH
# --
$ oedit /etc/rdz/tso2/ISPF.conf
-> change: change as needed
```

前の例のコマンドは、変更が必要な Developer for System z 構成ファイルを、新規に作成された tso2 ディレクトリーにコピーします。rsed.envvars 内の CGI_ISPCONF 変数を更新して新しい ISPF.conf ホーム・ディレクトリーを定義し、daemon.log および user.log を更新して新しいログ・ロケーション (これは、存在しなければ自動的に作成されます) を定義する必要があります。_RSE_RSED_PORT の更新により、既存の RSE デーモンと新規の RSE デーモンの両方が確実に固有のポ

ート番号を使用するようになります。CLASSPATH の更新により、tso2 にコピーされなかった構成ファイルを RSE が確実に検出できるようにします。ISPF.conf ファイル自体を、必要性に合わせて更新できます。ISPF 作業域 (rsed.envvars 内の変数 CGI_ISPWORK) を両方のインスタンスで共用できることに注意してください。

この時点で残っている作業は、新しいポート番号と新しい /etc/rdz/tso2 構成ファイルを使用する RSE 用の新規開始タスクを作成することです。rsed.envvars で _RSE_RSED_PORT が変更されていない場合、新規開始タスクは始動引数として、新規のポートを指定する必要がある点に注意してください。

このセクションで前述したアクションについて詳しくは、「*IBM Rational Developer for System z* ホスト構成ガイド」(SC88-5663) を参照してください。

第 11 章 実行、複数のインスタンスの

同じシステム上で Developer for System z の複数のインスタンスをアクティブにしたい場合があります。例えば、アップグレードをテストするときなどです。しかし、TCP/IP ポートなど、一部のリソースは共用できないため、デフォルトが常に適用可能であるとは限りません。この付録の情報を使用して Developer for System z の異なるインスタンスの共存を計画してください。その後、この構成ガイドを使用して、それらのインスタンスをカスタマイズできます。

Developer for System z の特定の部分は、2 つ (以上) のインスタンスで共用が可能です。それらのソフトウェア・レベルが同一で、しかも変更点が構成メンバー内に限られている場合を除いて、共用はお勧めできません。Developer for System z には、オーバーラップしない複数のインスタンスを作成するためのカスタマイズの余地が十分に残されており、それらのフィーチャーを使用することを強くお勧めします。

注:

- FEK および /usr/lpp/rdz は、製品のインストール時に使用された高位修飾子およびパスです。FEK.#CUST、/etc/rdz、および /var/rdz は、製品のカスタマイズ時に使用されたデフォルトのロケーションです (詳細については、「ホスト構成ガイド」 (SC88-5663) の「『カスタマイズ・セットアップ』」を参照してください)。
- 製品の z/OS UNIX の部分をデプロイしやすくするために、Developer for System z を専用ファイル・システム (HFS または zFS) にインストールしてください。
- 専用ファイル・システムを使用できない場合は、z/OS UNIX の tar コマンドなどのアーカイブ・ツールを使用して z/OS UNIX ディレクトリーをシステム間で転送してください。これにより、Developer for System z のファイルとディレクトリーの属性 (プログラム制御など) が保存されます。

Developer for System z インストール・ディレクトリーをアーカイブおよび復元するための以下のサンプル・コマンドについて詳しくは、「UNIX System Services コマンド解説書」 (SA88-8641) を参照してください。

- アーカイブ: `cd /SYS1/usr/lpp/rdz; tar -cSf /u/userid/rdz.tar`
- 復元: `cd /SYS2/usr/lpp/rdz; tar -xSf /u/userid/rdz.tar`

同一セットアップ、シスプレックス全体

Developer for System z の構成ファイル (およびコード) は、シスプレックス内の複数のシステム間で共用でき、いくつかのガイドラインに従っていれば、各システムが Developer for System z の同一コピーをそれぞれに保持して実行することができます。この情報がスタンドアロンの Developer for System z インスタンスに関するものであることに注意してください。Distributed Dynamic VIPA を使用してそれぞれ別々のシステムにある複数のサーバーを 1 つの仮想サーバーにグループ化するとき、TCP/IP セットアップの追加の規則が適用されます。70 ページの『Distributed Dynamic VIPA』に説明があります。

- ログ・ファイルを固有のロケーションに配置して、あるシステムが別のシステムの情報を上書きしないようにする必要があります。システム固有の z/OS UNIX ファイル・システムを指定のパスにマウントする場合は、rsed.envvars 内の daemon.log および user.log ディレクティブを使用して z/OS UNIX のログを特定のロケーションに送付することで、構成ファイルを共用できます。こうすると、すべてのログが論理的には同じ場所に書き込まれますが、その下のファイル・システムが共用されていないので、物理的には別々のロケーションに置かれることになります。
- /etc/rdz/ や /var/rdz/pushtoclient/ のような構成タイプのディレクトリーは、Developer for System z が読み取り専用モードで使用するため、シスプレックス全体で共用できます。
- 一時ファイル名はシスプレックスで認識されないため、/tmp/ や /var/rdz/WORKAREA/ のような一時データ・ディレクトリーは、システムごとに固有であることが必要です。
- コードを共用する場合は、保守適用後に同期のとれていないシステムが残らないように、構成ファイルも共用する必要があります。
- アクティブな /etc/rdz/pushtoclient.properties 構成ファイルを共用する場合は、関連するメタデータ・ディレクトリー /var/rdz/pushtoclient/ も共用する必要があります。

同一のソフトウェア・レベル、異なる構成ファイル

限定された一連の環境においては、カスタマイズ可能な部分 (の一部) を除くすべてを共用できます。1 つの例は、オンサイト使用のために非 SSL アクセスを提供し、オフサイト使用のために SSL エンコード通信を提供することです。

重要: 共用セットアップは、保守、テクニカル・プレビュー、または新規リリースのテストには、安全に使用できません。

アクティブな Developer for System z インストール済み環境の別のインスタンスをセットアップするには、現行セットアップとのオーバーラップを避けるため、異なるデータ・セット、ディレクトリー、およびポートを使用して、異なる部分のカスタマイズ・ステップを再実行します。

上記の SSL サンプルでは、現行 RSE デーモンのセットアップのクローンを作成でき、その後、クローンのセットアップを更新できます。次に、RSE デーモン始動 JCL のクローンを作成し、新しい TCP/IP ポートと更新した構成ファイルのロケーションを使用してカスタマイズすることができます。MVS カスタマイズ (JES ジョブ・モニターなど) は、SSL インスタンスと非 SSL インスタンスの間で共用できます。結果として、アクションは以下のようになります。

```
$ cd /etc/rdz
$ mkdir /etc/rdz/ssl
$ cp rsed.envvars /etc/rdz/ssl
$ cp ssl.properties /etc/rdz/ssl
$ ls /etc/rdz/ssl/
rsed.envvars    ssl.properties
$ oedit /etc/rdz/ssl/rsed.envvars
-> change: _RSE_RSED_PORT=4047
-> change: -Ddaemon.Log=/var/rdz/logs/ssl
```

```

-> change: -Duser.log=/var/rdz/logs/ssl
-> add at the END:
# -- NEEDED TO FIND THE REMAINING CONFIGURATION FILES
CFG_BASE=/etc/rdz
CLASSPATH=.:$CFG_BASE:$CLASSPATH
# --
$ oedit /etc/rdz/ssl/ssl.properties
-> change: change as needed

```

上記のコマンドは、変更が必要な Developer for System z 構成ファイルを、新規に作成された ssl ディレクトリにコピーします。rsed.envvars 内の daemon.log および user.log 変数を更新して、新しいログ・ロケーション (これは、存在しなければ自動的に作成されます) を定義する必要があります。CLASSPATH の更新により、ssl にコピーされなかった構成ファイルを RSE が確実に検出できるようにします。ssl.properties ファイル自体を、必要性に合わせて更新できます。

この時点で残っている作業は、新しいポート番号と新しい /etc/rdz/ssl 構成ファイルを使用する RSE 用の新規開始タスクを作成することです。

このセクションで前述したアクションについて詳しくは、「*IBM Rational Developer for System z* ホスト構成ガイド」(SC88-5663) の関連セクションを参照してください。

自動同期

先述の SSL サンプルでは、非 SSL と SSL 使用可能 RSE デーモンの違いはわずかであり、rsed.envvars ファイルの同期状態を保つためのプロセスを自動化することが可能です。こうすることで、rsed.envvars ファイル 1 つだけを維持すればよいので、サービスのロールアウトがシンプルになります。

以下の例は、RSED ポート番号をログ・ディレクトリ名に追加し、CLASSPATH を更新することによって、複製が残りの構成ファイルを検出します。さらに、この例では、SSL 使用可能 RSE デーモンの開始タスク JCL を拡張して、始動時に非 SSL の RSE デーモンの rsed.envvars を複製して、プロセスのポート番号を更新します。ポート番号はログ・ディレクトリ名に組み込まれるため、両方のデーモンの間で自動的に差異が生じることになります。

1. マスター rsed.envvars を準備します。

```

$ oedit /etc/rdz/rsed.envvars
-> change: -Ddaemon.log=/var/rdz/logs/$RSE_RSED_PORT
-> change: -Duser.log=/var/rdz/logs/$RSE_RSED_PORT
-> add at the END:
# -- NEEDED BY CLONES TO FIND THE REMAINING CONFIGURATION FILES
CFG_BASE=/etc/rdz
CLASSPATH=.:$CFG_BASE:$CLASSPATH
# --

```

2. 他の構成ファイル (rsed.envvars ファイル以外) を準備します。これは、マスター (非 SSL) と複製 (SSL) で異なります。

```

$ mkdir /etc/rdz/ssl
$ cp /etc/rdz/ssl.properties /etc/rdz/etc/rdz/ssl
$ oedit /etc/rdz/ssl/ssl.properties
-> change: change as needed

```

3. RSED 開始タスクを作成します。これは、基本 rsed.envvars を複製し、RSE デーモン・ポートを変更します (4035 -> 4034)。

```

/*
/* RSE DAEMON - SSL
/*
//RSED      PROC IVP=,                * 'IVP' to do an IVP test
//          HOME='/usr/lpp/rdz',
//          CNFG='/etc/rdz/ssl'
/*
//          SET SED='"/RSED_PORT/s/4035/4034/'
//          SET FILE='rsed.envvars'
/*
/* copy /etc/rdz/rsed.envvars to /etc/rdz/ssl/rsed.envvars
/* and alter RSED_PORT
/*
//CLONE     EXEC PGM=BPXBATCH,REGION=0M,COND=(4,LT),
// PARM='SH cd &CNFG;sed &SED ../&FILE>&FILE'
//STDOUT    DD SYSOUT=*
//STDERR    DD SYSOUT=*
/*
/* start RSED with the newly created rsed.envvars
/*
//RSED      EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,COND=(4,LT),
// PARM='PGM &HOME./bin/rsed.sh &IVP -C&CNFG'
//STDOUT    DD SYSOUT=*
//STDERR    DD SYSOUT=*
//          PEND
/*

```

その他のすべての状態

コードの変更が関与する場合（保守、テクニカル・プレビュー、新規リリース）、または変更がかなり複雑な場合は、Developer for System z の別のインストールを実行する必要があります。ここでは、さまざまなインストール間で発生する可能性がある競合点について説明します。

次のリストは、Developer for System z の複数のインスタンス間で異なるものにする必要があるか、そうすることが強く推奨される項目の要約です。

- SMP/E CSI
- インストール・ライブラリー
- JES ジョブ・モニター TCP/IP ポート、およびその構成ファイル FEJJCNFG
- JES ジョブ・モニター始動 JCL
- APPC トランザクション名
- RSE 構成ファイル、rsed.envvars、*.properties、および *.conf
- RSE TCP/IP ポート
- RSE 始動 JCL

以下で、これらの概要を説明します。

- SMP/E CSI
 1. Developer for System z の各インスタンスを別々の CSI にインストールします。SMP/E は同じ FMID が 1 つの CSI に再度インストールされるのを阻止しますが、別の FMID のインストールは受け入れます。2 番目の FMID の方が新しいバージョンであれば、既存のバージョンの製品は削除されます。2 番目の FMID の方が古いバージョンであれば、インストールはパーツ名の重複のために失敗します。
- インストール・ライブラリー

1. Developer for System z の各インスタンスを別々のデータ・セットおよびディレクトリーにインストールします。留意すべき点は、IBM 提供のデフォルトの /usr/lpp/rdz にプレフィックスを付けることによって、z/OS UNIX パスだけを変更できるということです。妥当な例は、/service/usr/lpp/rdz です。
 2. カスタマイズ・セットアップ・ジョブ FEK.SFEKSAMP(FEKSETUP) は、構成ファイルを保管するために使用するデータ・セットとディレクトリーを作成します。構成ファイルは固有でなければならず、既存のカスタマイズを上書きしないようにするために、このジョブを実行依頼するときは、固有のデータ・セット名とディレクトリー名を使用する必要があります。
- 必須部分
 1. JES ジョブ・モニター構成ファイル FEK.#CUST.PARMLIB(FEJJCNFG) は、JES ジョブ・モニターの TCP/IP ポート番号を保持しているので、共用することはできません。このメンバー自体は名前を変更できるので (JCL も更新する場合)、インストール・データ・セット内で更新を行うのでない場合は、このメンバーのカスタマイズ済みバージョンをすべて、同じデータ・セット内に配置できます。
 2. JES ジョブ・モニター始動 JCL FEK.#CUST.PROCLIB(JMON) は FEJJCNFG を参照し、したがって、これも共用することはできません。メンバー (および、ユーザー・ジョブとして始動する場合は JOB カード) を名前変更した後、すべての JCL を同じデータ・セット内に配置できます。
 3. RSE 構成ファイル /etc/rdz/rsed.envvars は、インストール・パスへの参照を保持し、オプションとして、サーバー・ログ・ロケーション (これは固有であることが必要です) への参照も保持します。このファイル名は必須なので、同じディレクトリー内にさまざまなコピーを保持することはできません。
 4. ISPF.conf 構成ファイルは FEK.SFEKPROC(FEKFRSRV)、つまり TSO コマンド・サーバーを参照しています。これはソフトウェア・レベル固有なので、インスタンスごとに ISPF.conf ファイルを作成する必要があります。
 5. 上記以外のすべての z/OS UNIX ベースの構成ファイル (*.properties など) は、rsed.envvars と同じディレクトリーに存在する必要があるため、したがって、共用できません。rsed.envvars は非共用ロケーションに置かれている必要があるからです。
 6. RSE 始動 JCL の FEK.#CUST.PROCLIB(RSED) は TCP/IP ポート番号を定義しており、固有でなければならないインストールおよび構成ディレクトリーを参照しているため、共用できません。メンバー (および、ユーザー・ジョブとして始動する場合は JOB カード) を名前変更した後、すべての JCL を同じデータ・セット内に配置できます。
 - オプションのパーツ
 1. REXEC ポートおよび SSH TCP/IP ポートは、無制限に共用できます。
 2. APPC トランザクションは FEK.SFEKPROC(FEKFRSRV)、つまり TSO コマンド・サーバーを参照しています。これはソフトウェア・レベル固有なので、インスタンスごとに APPC トランザクションを作成する必要があります。APPC トランザクション名は変化するので、rsed.envvars 内で _FEKFCMD_TP_NAME_ 変数を定義する必要があることに留意してください。

3. 一部の ELAXF* プロシージャーは hlq.SFEKLOAD、つまり Developer for System z のロード・ライブラリーを参照しています。さまざまなセットをユーザーが使用できるようにするために考えられるソリューションは、「*ホスト構成ガイド*」(SC88-5663) の"『ELAXF* リモート・ビルド・プロシージャー』" で JCLLIB についての注を参照してください。
4. DB2® ストアード・プロシージャーの 2 つのインスタンスをアクティブにするには、以下の作業を完了する必要があります。ただし、この説明はサポートの対象ではなく、保証がないことに注意してください。
 - a. hlq.SFEKPROC(ELAXMREX) を、別の名前を付けたメンバー、例えば ELAXMRXX にコピーします。
 - b. サンプル・メンバー hlq.SFEKSAMP(ELAXMSAM) を、別の名前を付けたメンバー、例えば ELAXMWDZ にコピーします。
 - c. これらの名前変更が反映されるように、サンプル・メンバー hlq.SFEKSAMP(ELAXMJCL) を変更します。以下に例を示します。

```
//SYSIN DD *
CREATE PROCEDURE SYSPROC.ELAXMRXX
  ( IN FUNCTION_REQUEST  VARCHAR(20)          CCSID EBCDIC
  ...
  , OUT RETURN_VALUE     VARCHAR(255)         CCSID EBCDIC )
PARAMETER STYLE GENERAL RESULT SETS 1
LANGUAGE REXX
COLLID DSNREXCS          WLM ENVIRONMENT ELAXMWDZ
PROGRAM TYPE MAIN        MODIFIES SQL DATA
STAY RESIDENT NO         COMMIT ON RETURN NO
ASUTIME NO LIMIT         SECURITY USER;

COMMENT ON PROCEDURE SYSPROC.ELAXMRXX IS
  'PLI & COBOL PROCEDURE PROCESSOR (ELAXMRXX), INTERFACE LEVEL 0.01';

GRANT EXECUTE ON PROCEDURE SYSPROC.ELAXMRXX TO PUBLIC;
//
```

- d. 「*ホスト構成ガイド*」(SC88-5663) の"『(オプション) DB2 ストアード・プロシージャー』"の説明に従って (ただし、新規メンバーを使用して) カスタマイズを続行します。
 - e. 新しい WLM 環境名 (例えば、ELAXMWDZ) を、クライアント上の DB2 ストアード・プロシージャー・ウィザードで使用する必要があります。
5. CICS 領域での Bidi サポートはロード・ライブラリー・メンバーに依存しているため、複数のリリースにまたがって共用することはできません。しかし、そのロード・モジュール名がすべてのインスタンスで同一の場合は、複数のリリースにまたがっても最新バージョンをインスタンス間で共用できます。ロード・モジュールの名前が変更された場合は、後方互換性を利用できません。
 6. CICS 領域に組み込まれている Application Deployment Manager ロード・モジュールには後方互換性があるので、最新バージョンを複数のリリースにまたがって共用できます。
 7. Application Deployment Manager の CRD VSAM には後方互換性があるので、最新バージョンを複数のリリースにまたがって共用できます。
 8. Application Deployment Manager の CICS リソース定義には後方互換性があるので、最新バージョンを複数のリリースにまたがって共用できます。
 9. CARMA VSAM はソフトウェア・レベル間で変更されている可能性があるもので、共用はお勧めできません。

第 12 章 トラブルシューティング、構成問題の

この章は、Developer for System z の構成時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのもので、以下のセクションで構成されています。

- 184 ページの『ログとセットアップの分析に使用、FEKLOGS』
- 184 ページの『ログ・ファイル』
- 190 ページの『ダンプ・ファイル』
- 193 ページの『トレース』
- 196 ページの『z/OS UNIX 許可ビット』
- 199 ページの『予約済み TCP/IP ポート』
- 201 ページの『アドレス・スペース・サイズ』
- 202 ページの『各種情報』

資料「Developer for System z メッセージとコード・ガイド」(SA88-4565) には、Developer for System z のコンポーネントが生成するメッセージおよび戻りコードについて記載されています。Developer for System z Answers to common host configuration and maintenance issues (SC14-7373) では、様々な問題のシナリオと、その解決方法が記載されています。

詳細は、Developer for System z Web サイト (<http://www-03.ibm.com/software/products/us/en/developerforsystemz/>) の Support セクションで入手できます。このサイトには、弊社のサポート・チームからの最新情報をもたらし技術情報が掲載されています。

この Web サイト (<http://www-01.ibm.com/support/docview.wss?uid=swg27038517>) のライブラリー・セクションには、Developer for System z 資料の最新バージョンが、ホワイト・ペーパーも含めて掲載されています。

Developer for System z インフォメーション・センター (<http://pic.dhe.ibm.com/infocenter/ratdevz/v9r0/index.jsp>) では、Developer for System z クライアント、およびそのクライアントとホストとの対話の方法が (クライアントの視点から) 説明されています。

また、z/OS インターネット・ライブラリー (<http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/>) にも有益な情報があります。

Developer for System z に不足している機能があると思われる場合はお知らせください。機能拡張要求 (RFE) は、次の場所で開示することができます。

<https://www.ibm.com/developerworks/support/rational/rfe/>

ログとセットアップの分析に使用、FEKLOGS

Developer for System z には、z/OS UNIX の全ログ・ファイル、および Developer for System z のインストールと構成に関する情報を収集するサンプル・ジョブ FEKLOGS が用意されています。

サンプル・ジョブ FEKLOGS は、FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳しくは、「ホスト構成ガイド」(SC88-5663) の"『カスタマイズ・セットアップ』"を参照してください。

FEKLOGS のカスタマイズは、JCL 内で記述されています。このカスタマイズでは、いくつかの主要な変数を準備します。

注: SDSF をご使用のお客様は、SDSF で **XDC** 行コマンドを使用してジョブ出力をデータ・セットに保存し、それを IBM サポートに提供することもできます。レコードの切り捨てを防ぐため、出力データ・セットは VB 2051 (デフォルト値は SDSF で VB 240) として割り当てられる必要があることに注意してください。

ログ・ファイル

Developer for System z は、お客様と IBM サポートによる問題の識別と解決に役立つログ・ファイルを作成します。次のリストは、z/OS ホスト・システム上に作成できるログ・ファイルの概要です。これらの製品固有のログとは別に、SYSLOG に関連するメッセージがないかどうか、必ず確認してください。

MVS ベースのログは、該当する DD ステートメントによって見つけることができます。z/OS UNIX ベースのログ・ファイルは、以下のディレクトリーに置かれます。

- userlog/\$LOGNAME/

ユーザー固有のログ・ファイルは、userlog/\$LOGNAME に置かれます。ここで、userlog は rsed.envvars 内の user.log ディレクティブと DSTORE_LOG_DIRECTORY ディレクティブを組み合わせた値で、\$LOGNAME はログオン・ユーザー ID (大文字) です。user.log ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。DSTORE_LOG_DIRECTORY ディレクティブがコメント化されているか存在しない場合は、user.log の値に .eclipse/RSE/ が付加されます。

- .dstoreMemLogging - DataStore メモリー使用率ロギング
- .dstoreTrace - DataStore アクション・ロギング
- .dstoreHashmap.* - アクティブ DataStore ハッシュ・マップのスナップショット
- .dstoreStackTrace.* - アクティブな DataStore スレッドとそれが呼び出された場所のスナップショット
- ffs.log - ネイティブの MVS 機能を実行する Foreign File System (FFS) サーバーのログ

- `ffsget.log` - 順次データ・セットまたは PDS メンバーを読み取るファイル読み取りプログラムのログ
- `ffsput.log` - 順次データ・セットまたは PDS メンバーを書き込むファイル書き込みプログラムのログ
- `lock.log` - 順次データ・セットまたは PDS メンバーをロック/ロック解除するロック・マネージャーのログ
- `rmt_class_loader.cache.jar` - RSE リモート・クラス・ローダーによってロードされたクラスのキャッシュ
- `rsecomm.log` - クライアントからのコマンドを処理する RSE サーバーのログ、および RSE に依存しているすべてのサービスの通信ロギング (Java 例外スタック・トレースを含んでいる場合がある)

注:

- `.eclipse` ディレクトリーおよび `.dstore*` ログ・ファイルはドット (.) で始まります。これにより、それらは非表示になります。非表示のファイルおよびディレクトリーをリストするには、z/OS UNIX コマンドの `ls -lA` を使用します。Developer for System z クライアントを使用している場合は、「ウィンドウ」>「設定...」>「リモート・システム」>「ファイル」設定ページを選択し、「隠しファイルの表示」を有効にします。

• `daemon-home`

RSE デーモンおよび RSE スレッド・プールに固有のログ・ファイルは、`daemon-home` に置かれます。ここで、`daemon-home` は `rsed.envvars` 内の `daemon.log` ディレクティブの値になります。`daemon.log` ディレクティブがコメント化されているか存在しない場合は、RSED 開始タスクに割り当てられているユーザー ID のホーム・ディレクトリーが使用されます。ホーム・ディレクトリーはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

- `rsedaemon.log` - RSE デーモンのログ
- `rseserver.log` - RSE スレッド・プールのログ
- `audit.log` - RSE 監査証跡
- `serverlogs.count` - RSE スレッド・プール・ストリームをログに記録するためのカウンター
- `stderr*.log` - RSE スレッド・プールの標準エラー・ストリーム
- `stdout*.log` - RSE スレッド・プールの標準出力ストリーム

• `/tmp`

この変数が `rsed.envvars` で定義されている場合は、IVP 固有のログ・ファイル (インストール検査プログラム) は、`TMPDIR` で参照されるディレクトリーに置かれます。この変数が定義されない場合は、そのファイルは `/tmp` に作成されます。

- `fekfivpi.log` - `fekfivpi` IVP テストのログ
- `fekfivps.log` - `fekfivps` IVP テストのログ
- `fekfivpc.log` - `fekfivpc` IVP テストの通信ログ

注: オペレーター・コマンドを使用して、上記のいくつかのログ・ファイルにデータを書き込まれる量を制御できます。詳しくは、「[ホスト構成ガイド](#)」(SC88-5663) の『オペレーター・コマンド』を参照してください。

JES ジョブ・モニター・ロギング

• SYSOUT DD

通常操作のロギング。サンプル JCL FEK.#CUST.PROCLIB(JMON) 内のデフォルト値は SYSOUT=* です。

• SYSPRINT DD

トレース・ロギング。サンプル JCL FEK.#CUST.PROCLIB(JMON) 内のデフォルト値は SYSOUT=* です。トレースは -TV パラメーターでアクティブにされます。詳細については、193 ページの『JES ジョブ・モニターのトレース』を参照してください。

RSE デーモンおよびスレッド・プールのロギング

• STDOUT DD

RSE デーモンの Java 標準出力 stdout のリダイレクトされたデータ。サンプル JCL FEK.#CUST.PROCLIB(RSED) 内のデフォルト値は SYSOUT=* です。

• STDERR DD

RSE デーモンの Java 標準エラー出力 stderr のリダイレクトされたデータ。サンプル JCL FEK.#CUST.PROCLIB(RSED) 内のデフォルト値は SYSOUT=* です。

• daemon-home

RSE デーモンおよび RSE スレッド・プールに固有のログ・ファイルは、daemon-home に置かれます。ここで、daemon-home は rsed.envvars 内の daemon.log ディレクティブの値になります。daemon.log ディレクティブがコメント化されているか存在しない場合は、RSED 開始タスクに割り当てられているユーザー ID のホーム・ディレクトリーが使用されます。ホーム・ディレクトリーはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

- rsedaemon.log - RSE デーモンのログ
- rseserver.log - RSE スレッド・プールのログ
- audit.log - RSE 監査証跡
- serverlogs.count - RSE スレッド・プール・ストリームをログに記録するためのカウンター
- stderr.*.log - RSE スレッド・プールの標準エラー・ストリーム
- stdout.*.log - RSE スレッド・プールの標準出力ストリーム

注:

- serverlogs.count、stderr.*.log、および stdout.*.log は、rsed.envvars 内の enable.standard.log ディレクティブがアクティブである場合、またはこの機能が **modify rsestandardlog on** オペレーター・コマンドで動的にアクティブ化される場合にのみ作成されます。
- stderr.*.log および stdout.*.log 内の * は、デフォルトでは 1 です。ただし、複数の RSE スレッド・プールが存在することが可能であるため、ファイル名が固有となるように、新しい RSE スレッド・プールが増えるたびにこの数値に 1 が加算されます。

- オペレーター・コマンドを使用して、上記のいくつかのログ・ファイルにデータを書き込まれる量を制御できます。詳しくは、「[ホスト構成ガイド](#)」(SC88-5663)の『オペレーター・コマンド』を参照してください。
- `rsed.envvars` で `keep.last.log=true` が指定されている場合、`rse*.log` ファイルは「.log」拡張子の代わりに「.last」拡張子を使用することもできます。デフォルトでは、「.last」ログ・ファイルは作成されません。
- `rsed.envvars` で `keep.all.logs=true` が指定されている場合、`rse*.log` ファイルは拡張名を持ちます。デフォルトで、拡張名が使用されます。以下に拡張名の例を示します。RSED は、RSE デーモンのアドレス・スペースの名前を、`yyyymmddhhmmss` は、日時のタイム・スタンプ (年、月、日、時、分、秒) を表します: `rseserver.RSED#yyyymmddhhmmss.log`

RSE ユーザー・ロギング

• `userlog/$LOGNAME/`

RSE に関連するコンポーネントによって作成される複数のログ・ファイルがあります。これらはいずれも `userlog/$LOGNAME` に置かれます。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブと `DSTORE_LOG_DIRECTORY` ディレクティブを組み合わせた値で、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。`DSTORE_LOG_DIRECTORY` ディレクティブがコメント化されているか存在しない場合は、`user.log` の値に `.eclipse/RSE/` が付加されます。

- `.dstoreMemLogging` - DataStore メモリー使用率ロギング
- `.dstoreTrace` - DataStore アクション・ロギング
- `.dstoreHashmap.*` - アクティブ DataStore ハッシュ・マップのスナップショット
- `.dstoreStackTrace.*` - アクティブな DataStore スレッドとそれが呼び出された場所のスナップショット
- `ffs.log` - ネイティブの MVS 機能を実行する Foreign File System (FFS) サーバーのログ
- `ffsget.log` - 順次データ・セットまたは PDS メンバーを読み取るファイル読み取りプログラムのログ
- `ffsput.log` - 順次データ・セットまたは PDS メンバーを書き込むファイル書き込みプログラムのログ
- `lock.log` - 順次データ・セットまたは PDS メンバーをロックまたはロック解除するロック・マネージャーのログ
- `rmt_class_loader.cache.jar` - RSE リモート・クラス・ローダーによってロードされたクラスのキャッシュ
- `rsecomm.log` - クライアントからのコマンドを処理する RSE サーバーのログ、および RSE に依存しているすべてのサービスの通信ロギング (Java 例外スタック・トレースを含んでいる場合がある)

注:

- `.eclipse` ディレクトリーおよび `.dstore*` ログ・ファイルはドット (.) で始まります。これにより、それらは非表示になります。非表示のファイルおよびディレクトリーをリストするには、`z/OS UNIX` コマンドの `ls -lA` を使用します。
Developer for System z クライアントを使用している場合は、「ウィンドウ」>「設定...」>「リモート・システム」>「ファイル」設定ページを選択し、「隠しファイルの表示」を有効にします。
- `.dstore*` ログ・ファイルの作成は、「ホスト構成ガイド」(SC88-5663) の「『_RSE_JAVAOPTS での追加 Java 始動パラメーター』」の説明にあるように、`-DDSTORE_* Java 始動オプション`によって制御されます。
- `.dstore*` ログ・ファイルは、ASCII で作成されます。これらのログ・ファイルは EBCDIC で表示する (コード・ページ IBM-1047 を使用している場合) には、`z/OS UNIX` コマンドの `iconv -f ISO8859-1 -t IBM-1047 .dstore*` を使用します。
- すべての `*.log` ファイルとは異なり、`.dstore*` ログ・ファイルはクライアントの再接続の際に自動的に削除されません。これらのファイルの削除は、手動で操作します。
- オペレーター・コマンドを使用して、上記のいくつかのログ・ファイルにデータを書き込まれる量を制御できます。詳しくは、「ホスト構成ガイド」(SC88-5663)の『オペレーター・コマンド』を参照してください。
- `rsed.envvars` で `keep.last.log=true` が指定されている場合、`ffs*.log`、`lock.log` および `rsecomm.log` ファイルは「.log」拡張子の代わりに「.last」拡張子を使用することもできます。デフォルトでは「.last」ログ・ファイルは作成されません。
- `rsed.envvars` で `keep.all.logs=true` が指定されている場合、`ffs*.log`、`lock.log` および `rsecomm.log` ファイルは拡張名を持ちます。デフォルトで、拡張名が使用されます。以下に拡張名の例を示します。`RSEDx` は、ユーザーがアクティブになっているスレッド・プールのアドレス・スペースの名前を、`yyyymmddhhmmss` は、日時のタイム・スタンプ (年、月、日、時、分、秒) を表します：`ffs.RSEDx#yyyymmddhhmmss.log`

SCLM Developer Toolkit のロギング

- `userlog/$LOGNAME/rsecomm.log`

SCLM Developer Toolkit の通信ロギング。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブと `DSTORE_LOG_DIRECTORY` ディレクティブを組み合わせた値で、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。`DSTORE_LOG_DIRECTORY` ディレクティブがコメント化されているか存在しない場合は、`user.log` の値に `.eclipse/RSE/` が付加されます。

CARMA ロギング

- CARMA サーバー・ジョブ

バッチ・インターフェースを使用して CARMA との接続を開くと、FEK.#CUST.SYSPROC(CRASUBMT) は CRAport というサーバー・ジョブを (ユーザーの、所有者としてのユーザー ID を使用して) 開始します。ここで、port は使用される TCP/IP ポートです。

- **CARMALOG DD**

選択された CARMA 始動方式で DD ステートメント CARMALOG が指定されている場合、CARMA ロギングはサーバー・ジョブでその DD ステートメントへリダイレクトされ、指定されていない場合は SYSPRINT へ送られます。

- **SYSPRINT DD**

サーバー・ジョブの SYSPRINT DD は、DD ステートメント CARMALOG が定義されていない場合、CARMA ロギングを保持します。

- **SYSTSPRT DD**

サーバー・ジョブの SYSTSPRT DD は、CARMA サーバー始動のシステム (TSO) メッセージを保持します。

- **userlog/\$LOGNAME/rsecomm.log**

CARMA の通信ロギング。ここで、userlog は rsed.envvars 内の user.log ディレクティブと DSTORE_LOG_DIRECTORY ディレクティブを組み合わせた値で、\$LOGNAME はログオン・ユーザー ID (大文字) です。user.log ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。DSTORE_LOG_DIRECTORY ディレクティブがコメント化されているか存在しない場合は、user.log の値に .eclipse/RSE/ が付加されます。

fekfivpc IVP テスト・ロギング

- **/tmp/fekfivpc.log**

fekfivpc コマンド (CARMA 関連の IVP テスト) は、fekfivpc.log ファイルを作成して、RSE と CARMA の間の通信を文書化します。この変数が rsed.envvars で定義されている場合は、ログは TMPDIR で参照されるディレクトリーに作成されます。この変数が定義されない場合は、そのファイルは /tmp に作成されます。

fekfivpi IVP テスト・ロギング

- **/tmp/fekfivpi.log**

fekfivpi -file コマンドの出力 (TSO/ISPF クライアント・ゲートウェイ関連の IVP テスト)。この変数が rsed.envvars で定義されている場合は、ログは TMPDIR で参照されるディレクトリーに作成されます。この変数が定義されない場合は、そのファイルは /tmp に作成されます。

fekfivps IVP テスト・ロギング

- **/tmp/fekfivps.log**

febfivps -file コマンドの出力 (SCLMDT 関連の IVP テスト)。この変数が rsed.envvars で定義されている場合は、ログは TMPDIR で参照されるディレクトリに作成されます。この変数が定義されない場合は、そのファイルは /tmp に作成されます。

コード・レビューのロギング

- SYSTSPRT DD

コード・レビュー・プロシーチャーを呼び出すステップの SYSTSPRT DD は、コード分析プロセスを開始する、フロントエンドのメッセージを保持します。

- WORKSPCE DD

コード・レビュー・プロシーチャーを呼び出すステップの WORKSPCE DD は、コード分析プロセスの Eclipse ワークスペース・ログ・メッセージを保持します。

- ERRMSGs DD

コード・レビュー・プロシーチャーを呼び出すステップの ERRMSGs DD は、コード分析プロセスの stderr 出力を保持します。

コード・カバレッジのロギング

- SYSTSPRT DD

コード・レビュー・プロシーチャーを呼び出すステップの SYSTSPRT DD は、コード分析プロセスを開始する、フロントエンドのメッセージを保持します。

- WORKSPCE DD

コード・レビュー・プロシーチャーを呼び出すステップの WORKSPCE DD は、コード分析プロセスの Eclipse ワークスペース・ログ・メッセージを保持します。

- ERRMSGs DD

コード・レビュー・プロシーチャーを呼び出すステップの ERRMSGs DD は、コード分析プロセスの stderr 出力を保持します。

ダンプ・ファイル

製品が異常終了した場合、問題判別を支援するためにストレージ・ダンプが作成されます。これらのダンプの可用性とロケーションは、サイト固有の設定に大きく依存します。ダンプは作成されない場合や、次のセクションで述べるロケーションとは別のロケーションに作成される場合があります。

MVS ダンプ

プログラムを MVS 内で実行している場合は、システム・ダンプ・ファイルを検査し、JCL に以下の DD ステートメント (製品によって異なります) があるかどうかを確認してください。

- SYSABEND

- SYSMDUMP
- SYSUDUMP
- CEEDUMP
- SYSPRINT
- SYSOUT

これらの DD ステートメントの詳細については、「*MVS JCL 解説書*」(SA88-8569) および「*Language Environment デバッグ・ガイド*」(GA88-8548) を参照してください。

Java ダンプ

z/OS UNIX では、大部分の Developer for System z ダンプが Java 仮想マシン (JVM) によって制御されます。

JVM は、デフォルトでは初期化時にダンプ・エージェント・セット (SYSTDUMP と JAVADUMP) を作成します。このダンプ・エージェント・セットは、JAVA_DUMP_OPTS 環境変数を使用してオーバーライドでき、さらに、コマンド行で -Xdump を使用することによってオーバーライドできます。JVM コマンド行オプションは、rsed.envvars の _RSE_JAVAOPTS ディレクティブで定義されます。IBM サポートから指示された場合以外は、ダンプの設定を一切変更しないでください。

注: コマンド行の -Xdump:what オプションを使用すると、始動完了時にどのダンプ・エージェントが存在するかを決定できます。

生成できるダンプのタイプは、以下のとおりです。

SYSTDUMP

Java トランザクション・ダンプ。z/OS によって生成される不定形式のストレージ・ダンプ。

ダンプは、%uid.JVM.TDUMP.%job.D%y%m%d.T%H%M%S という書式のデフォルト名、または JAVA_DUMP_TDUMP_PATTERN 環境変数の設定によって決まるデフォルト名を使用して、順次 MVS データ・セットに書き込まれます。

注: JAVA_DUMP_TDUMP_PATTERN を使用すると変数を使用できます。それらの変数は、トランザクション・ダンプの取得時に実際の値に変換されます。

表 38. JAVA_DUMP_TDUMP_PATTERN 変数

変数	使用法
%uid	ユーザー ID
%job	ジョブ名
%y	年 (2 桁)
%m	月 (2 桁)
%d	日 (2 桁)
%H	時間 (2 桁)
%M	分 (2 桁)
%S	秒 (2 桁)

CEEDUMP

言語環境プログラム (LE) ダンプ。定様式の要約システム・ダンプ。これは JVM プロセス内の各スレッドのスタック・トレースを、レジスター情報や各レジスターのストレージの短いダンプと一緒に表示します。

ダンプは CEEDUMP.yyyyymmdd.hhmmss.pid という名前の z/OS UNIX ファイルに書き込まれます。ここで、yyyymmdd は現在の日付で、hhmmss は現在の時刻、pid は現行プロセス ID です。このファイルの可能なロケーションについては、『z/OS UNIX ダンプ・ロケーション』に説明があります。

HEAPDUMP

Java ヒープ上にあるオブジェクトの定様式ダンプ (リスト)。

ダンプは HEAPDUMP.yyyyymmdd.hhmmss.pid.TXT という名前の z/OS UNIX ファイルに書き込まれます。ここで、yyyymmdd は現在の日付で、hhmmss は現在の時刻、pid は現行プロセス ID です。このファイルの可能なロケーションについては、『z/OS UNIX ダンプ・ロケーション』に説明があります。

Developer for System z がオペレーター・コマンドを提供して、このダンプをトリガーすることに注意してください。詳しくは、「ホスト構成ガイド」(SC88-5663) の「オペレーター・コマンド」の章を参照してください。

JAVADUMP

JVM の定様式分析。これには JVM と Java アプリケーションに関連した診断情報 (例えば、アプリケーション環境、スレッド、ネイティブ・スタック、ロック、メモリーなど) が入っています。

ダンプは JAVADUMP.yyyyymmdd.hhmmss.pid.TXT という名前の z/OS UNIX ファイルに書き込まれます。ここで、yyyymmdd は現在の日付で、hhmmss は現在の時刻、pid は現行プロセス ID です。このファイルの可能なロケーションについては、『z/OS UNIX ダンプ・ロケーション』に説明があります。

Developer for System z がオペレーター・コマンドを提供して、このダンプをトリガーすることに注意してください。詳しくは、「ホスト構成ガイド」(SC88-5663) の「オペレーター・コマンド」の章を参照してください。

JVM ダンプについて詳しくは、「Java Diagnostic Guide」(SC34-6358) を、また、LE 固有の情報については、「Language Environment デバッグ・ガイド」(GA88-8548) を参照してください。

z/OS UNIX ダンプ・ロケーション

JVM は、以下の各ロケーションの有無と書き込み権限を検査し、最初に使用可能なロケーションに CEEDUMP、HEAPDUMP、および JAVADUMP ファイルを保管します。ダンプ・ファイルを正しく書き込むための十分なディスク・スペースが必要であることに注意してください。

1. 環境変数 _CEE_DMPTARG 内のディレクトリー (ある場合)。この変数は、rsed.envvars 内で /tmp として設定されます。これを /dev/null に変更して、ダンプ・ファイルが作成されないようにすることができます。

2. 現行作業ディレクトリー (このディレクトリーがルート・ディレクトリー (/) でなく、書き込み可能である場合)。
3. 環境変数 TMPDIR (一時ディレクトリーが /tmp でない場合に、そのロケーションを示す環境変数) 内のディレクトリー(ある場合)。
4. /tmp ディレクトリー。
5. 上記のロケーションのどこにも保管できない場合、ダンプは `stderr` に書き込まれます。

トレース

JES ジョブ・モニターのトレース

JES ジョブ・モニターのトレースは、「ホスト構成ガイド」(SC88-5663)の『オペレーター・コマンド』の説明にあるように、システム・オペレーターによって制御されます。

- `PRM=-TV` パラメーターを指定して `JMON` 開始タスクを始動すると、冗長モード (トレース) がアクティブになります。
- `modify trace` および `modify message` オペレーター・コマンドによって、ログ・メッセージの詳細レベルを選択できます。

RSE トレース

RSE に関連するコンポーネントによって作成される複数のログ・ファイルがあります。そのほとんどは `userlog/$LOGNAME` に置かれます。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブと `DSTORE_LOG_DIRECTORY` ディレクティブを組み合わせた値で、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。`DSTORE_LOG_DIRECTORY` ディレクティブがコメント化されているか存在しない場合は、`user.log` の値に `.eclipse/RSE/` が付加されます。

`ffs*.log`、`lock.log`、および `rsecomm.log` に書き込まれるデータの量は、`modify rsecommlog` オペレーター・コマンドによって制御するか、`rsecomm.properties` 内で `debug_level` を設定することによって制御します。詳しくは、「ホスト構成ガイド」(SC88-5663) の『オペレーター・コマンド』および「ホスト構成ガイド」(SC88-5663) の「『オプション』 RSE トレース」を参照してください。

`.dstore*` ログ・ファイルの作成は、「ホスト構成ガイド」(SC88-5663) の「『_RSE_JAVAOPTS』での追加 Java 始動パラメーター」の説明にあるように、`-DDSTORE_* Java` 始動オプションによって制御されます。

注:

- `.eclipse` ディレクトリーおよび `.dstore*` ログ・ファイルはドット (.) で始まります。これにより、それらは非表示になります。非表示のファイルおよびディレクトリーをリストするには、`z/OS UNIX` コマンドの `ls -lA` を使用します。

Developer for System z クライアントを使用している場合は、「ウィンドウ」>「設定...」>「リモート・システム」>「ファイル」設定ページを選択し、「隠しファイルの表示」を有効にします。

- **.dstore*** ログ・ファイルは、ASCII で作成されます。これらのログ・ファイルを EBCDIC で表示する (コード・ページ IBM-1047 を使用している場合) には、z/OS UNIX コマンドの **iconv -f ISO8859-1 -t IBM-1047 .dstore*** を使用します。
- すべての *.log ファイルとは異なり、**.dstore*** ログ・ファイルはクライアントの再接続の際に自動的に削除されません。これらのファイルの削除は、手動で操作します。

RSE デーモンおよび RSE スレッド・プールに固有のログ・ファイルは、daemon-home に置かれます。ここで、daemon-home は rsed.envvars 内の daemon.log ディレクティブの値になります。daemon.log ディレクティブがコメント化されているか存在しない場合は、RSED 開始タスクに割り当てられているユーザー ID のホーム・ディレクトリーが使用されます。ホーム・ディレクトリーはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

rsedaemon.log および rseserver.log に書き込まれるデータの量は、**modify rsedaemonlog** および **modify rseserverlog** オペレーター・コマンドによって制御するか、rsecomm.properties で debug_level を設定することによって制御します。詳しくは、「ホスト構成ガイド」(SC88-5663) の『オペレーター・コマンド』および「ホスト構成ガイド」(SC88-5663) の「『(オプション) RSE トレース』」を参照してください。

serverlogs.count、stderr.*.log、および stdout.*.log は、rsed.envvars 内の enable.standard.log ディレクティブがアクティブである場合、またはこの機能が **modify rsestandardlog on** オペレーター・コマンドで動的にアクティブ化される場合にのみ作成されます。

CARMA トレース

ユーザーは、クライアント上の CARMA 接続のプロパティー・タブで「トレース・レベル」を設定することにより、CARMA が生成するトレース情報の量を制御できます。「トレース・レベル」の選択項目は、以下のとおりです。

- ログギングを使用不可に設定
- エラー・ログギング
- 警告ログギング
- 通知ログギング
- デバッグ・ログギング

デフォルト値は以下のとおりです。

エラー・ログギング

ログ・ファイルのロケーションの詳細については、184 ページの『ログ・ファイル』を参照してください。

エラー・フィードバック・トレース

以下のプロシージャを使用すると、リモート・ビルド・プロシージャでのエラー・フィードバック問題を診断するために必要な情報を収集できます。このトレースはパフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。このセクションで、hlq と表記したものはすべて、Developer for System z のインストール時に使用した高位修飾子を指しています。インストールのデフォルトは FEK ですが、ご使用のサイトには当てはまらない場合があります。

1. アクティブな ELAXFCOC コンパイル・プロシージャのバックアップ・コピーを作成してください。このプロシージャは、デフォルトでは出荷時にデータ・セット hlq.SFEKSAMP に組み込まれていますが、「[ホスト構成ガイド](#)」(SC88-5663) の「『ELAXF* リモート・ビルド・プロシージャ』」で説明されているように、別のロケーション (SYS1.PROCLIB など) にコピーされている可能性があります。

2. アクティブな ELAXFCOC プロシージャを変更して、EXIT(ADEXIT(ELAXMGUX)) コンパイル・オプションに「MAXTRACE」ストリングを組み込みます。

```
//COBOL EXEC PGM=IGYCRCTL,REGION=2048K,
//*      PARM=('EXIT(ADEXIT(ELAXMGUX))',
//      PARM=('EXIT(ADEXIT('MAXTRACE',ELAXMGUX))',
//      'ADATA',
//      'LIB',
//      'TEST(NONE,SYM,SEP)',
//      'LIST',
//      'FLAG(I,I)'&CICS &DB2 &COMP)
```

注: MAXTRACE を二重のアポストロフィで囲む必要があります。このオプションは、次のようになります。EXIT(ADEXIT('MAXTRACE',ELAXMGUX))。

3. 詳細にトレースしたい COBOL プログラムに対し、リモート構文検査を実行します。
4. JES 出力の SYSOUT 部分は、SIDEFILE1、SIDEFILE2、SIDEFILE3、および SIDEFILE4 のデータ・セット名のリストで始まります。

```
ABOUT TOO OPEN SIDEFILE1 - NAME = 'uid.DT021207.TT110823.M0000045.C0000000'
SUCCESSFUL OPEN SIDEFILE1 - NAME = 'uid.DT021207.TT110823.M0000045.C0000000'
ABOUT TOO OPEN SIDEFILE2 - NAME = 'uid.DT021207.TT110823.M0000111.C0000001'
SUCCESSFUL OPEN SIDEFILE2 - NAME = 'uid.DT021207.TT110823.M0000111.C0000001'
ABOUT TOO OPEN SIDEFILE3 - NAME = 'uid.DT021207.TT110823.M0000174.C0000002'
SUCCESSFUL OPEN SIDEFILE3 - NAME = 'uid.DT021207.TT110823.M0000174.C0000002'
ABOUT TOO OPEN SIDEFILE4 - NAME = 'uid.DT021207.TT110823.M0000236.C0000003'
SUCCESSFUL OPEN SIDEFILE4 - NAME = 'uid.DT021207.TT110823.M0000236.C0000003'
```

注: 設定によっては、SIDEFILE1 および SIDEFILE2 が DD ステートメントを指している場合があります (SUCCESSFUL OPEN SIDEFILE1 - NAME = DD:WSEDSF1)。実際のデータ・セット名を得るには、出力の JESJCL 部分 (これは SYSOUT 部分の前に置かれます) を参照してください。

```
22 //COBOL.WSEDSF1 DD DISP=MOD,
// DSN=uid.ERRCOB.member.SF.Z682746.XML
23 //COBOL.WSEDSF2 DD DISP=MOD,
// DSN=uid.ERRCOB.member.SF.Z682747.XML
```

5. これら 4 つのデータ・セットを PC にコピーします。例えば、Developer for System z 内にローカル COBOL プロジェクトを作成し、SIDEFILE1->4 データ・セットを追加します。

6. 完全な JES ジョブ・ログを PC にコピーします。例えば、Developer for System z でジョブ出力を開き、それをローカル・プロジェクトに保存するために「ファイル」>「別名保存...」を選択します。
7. プロシーチャー ELAXFCOC を元の状態に復元します。これは、変更を元に戻す (コンパイル・オプション内の「MAXTRACE」ストリングを除去する) か、バックアップをリストアします。
8. 収集したファイル (SIDEFILE1->4 およびジョブ・ログ) を IBM サポートへ送ります。

z/OS UNIX 許可ビット

Developer for System z では、z/OS UNIX ファイル・システムおよび一部の z/OS UNIX ファイルに特定の許可ビットがセットされている必要があります。

SETUID ファイル・システム属性

リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供する Developer for System z コンポーネントです。これは、ユーザーのセキュリティ環境を作成するなどのタスクの実行を許可されている必要があります。

Developer for System z がインストールされているファイル・システム (HFS または zFS) は、SETUID 許可ビットをオン (これはデフォルトです) にしてマウントする必要があります。NOSETUID パラメーターを指定してファイル・システムをマウントすると、ユーザーのセキュリティ環境が Developer for System z によって作成されず、接続要求が失敗します。このセットアップの問題に関するその他の標識は以下のとおりです。

- コンソール・メッセージ「FEK999E The module, fekfomvs must be marked as APF-authorized (FEK999E モジュール fekfomvs は APF 許可としてマークを付ける必要があります)」
- パスチケット IVP は「ICH409I 282-010 ABEND DURING RACHECK PROCESSING (ICH409I 282-010 RACHECK 処理中に異常終了しました)」で失敗します。

Java または z/OS UNIX バイナリーをホスティングするファイル・システムが NOSETUID パラメーターを使って組み込まれている場合、類似したエラー (BPXP014I および BPXP015I というメッセージなど) が発生する可能性があります

TSO の **ISHELL** コマンドを使用して、SETUID ビットの現行ステータスをリストします。ISHELL パネルで、「File_systems」>「1. マウント・テーブル... (1. Mount table...)」を選択して、マウントされたファイル・システムをリストします。a 行コマンドは、選択されたファイル・システムの属性を表示します。ここで、「Ignore SETUID」フィールドは 0 であることが必要です。

プログラム制御許可

リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供する Developer for System z コンポーネントです。

これは、クライアントのユーザー ID への切り替えなどのタスクを実行するために、プログラム制御で実行する必要があります。

z/OS UNIX プログラム制御ビットは、SMP/E インストールのときに、必要ならば設定されます。ただし、21 ページの『第 2 章 セキュリティーに関する考慮事項』の説明にあるように、セキュリティ製品への Java インターフェースは除きます。この許可ビットは、Developer for System z ディレクトリーの手動コピー中に保存しなかった場合、失われることがあります。

以下の Developer for System z ファイルはプログラムで制御される必要があります。

- /usr/lpp/rdz/bin/
 - fekfdivp
 - fekfomvs
 - fekfrivp
- /usr/lpp/rdz/lib/
 - fekfdir.dll
 - libfekdcore.so
 - libfekfmain.so
- /usr/lpp/rdz/lib/icuc/
 - libicudata.dll
 - libicudata50.1.dll
 - libicudata50.dll
 - libicudata64.50.1.dll
 - libicudata64.50.dll
 - libicudata64.dll
 - libicuuc.dll
 - libicuuc50.1.dll
 - libicuuc50.dll
 - libicuuc64.50.1.dll
 - libicuuc64.50.dll
 - libicuuc64.dll

z/OS UNIX コマンド **ls -E** を使用して、拡張属性をリストします。このリストで、プログラム制御ビットには、次のサンプルに示すように英字の p のマークが付きます (\$ は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ ls -E lib/fekf*
-rwxr-xr-x -ps- 2 user      group      94208 Jul  8 12:31 lib/fekfdir.dll
```

プログラム制御ビットを手動でセットするには、次のサンプルに示すように、z/OS UNIX コマンド **extattr +p** を使用します (\$ および # は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ su
# extattr +p lib/fekf*
```

```
# exit
$ ls -E lib/fekf*
-rwxr-xr-x  -ps-  2 user      group      94208 Jul  8 12:31 lib/fekfdir.dll
```

注: **extattr +p** コマンドを使用するには、少なくとも、セキュリティー・ソフトウェアの FACILITY クラス内の BPX.FILEATTR.PROGCTL プロファイルに対する READ アクセス権が必要です。あるいは、このプロファイルが定義されていない場合は、スーパーユーザー (UID 0) であることが必要です。詳細については、「UNIX System Services 計画」(GA88-8639) を参照してください。

APF 許可

リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供する Developer for System z コンポーネントです。詳細なプロセス・リソース使用量を表示するなどのタスクを実行するためには、このコンポーネントを APF 許可がある状態で実行する必要があります。

z/OS UNIX APF ビットは、SMP/E インストールのときに、必要ならば設定されます。この許可ビットは、Developer for System z ディレクトリーの手動コピー中に保存しなかった場合、失われることがあります。

以下の Developer for System z ファイルには、APF 許可があることが必要です。

- /usr/lpp/rdz/bin/
 - BWBTSOW
 - CRASTART
 - fekfomvs
 - fekfriwp

拡張属性をリストするには、z/OS UNIX コマンド **ls -E** を使用します。このリストでは、次のサンプルに示すように、APF ビットに英字の **a** のマークが付きます (\$ は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ ls -E bin/fekfripv
-rwxr-xr-x  aps-  2 user      group      114688 Sep 17 06:41 bin/fekfripv
```

APF ビットを手動でセットするには、次のサンプルに示すように、z/OS UNIX コマンド **extattr +a** を使用します (\$ および # は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ su
# extattr +a bin/fekfripv
# exit
$ ls -E bin/fekfripv
-rwxr-xr-x  aps-  2 user      group      114688 Sep 17 06:41 bin/fekfripv
```

注: **extattr +a** コマンドを使用するには、少なくとも、セキュリティー・ソフトウェアの FACILITY クラス内の BPX.FILEATTR.APF プロファイルに対する READ アクセス権が必要です。あるいは、このプロファイルが定義されていない場合は、スーパーユーザー (UID 0) であることが必要です。詳細については、「UNIX System Services 計画」(GA88-8639) を参照してください。

スティッキー・ビット

オプションの Developer for System z サービスの中には、MVS ロード・モジュールを z/OS UNIX で使用可能にしなければならないものもあります。これを行うには、z/OS UNIX で、スティッキー・ビットをオンにしたスタブ (ダミー・ファイル) を作成します。スタブが実行されると、z/OS UNIX は同じ名前の MVS ロード・モジュールを探し、代わりにそのロード・モジュールを実行します。

z/OS UNIX スティッキー・ビットは、SMP/E インストールのときに、必要ならば設定されます。これらの許可ビットは、Developer for System z ディレクトリーの手動コピー中に保存しなかった場合、失われることがあります。

以下の Developer for System z ファイルは、スティッキー・ビットがオンであることが必要です。

- /usr/lpp/rdz/bin/
 - AZUTSTRN
 - BWBTSOW
 - BWBTRANT
 - CRASTART

z/OS UNIX コマンド **ls -l** を使用して、許可をリストします。このリストで、スティッキー・ビットには、次のサンプルに示すように英字の **t** のマークが付きます (\$ は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ ls -l bin/CRA*
-rwxr-xr-t  2 user      group          71 Jul  8 12:31 bin/CRASTART
```

スティッキー・ビットを手動でセットするには、次のサンプルに示すように、z/OS UNIX コマンド **chmod +t** を使用します (\$ および # は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ su
# chmod +t bin/CRA*
# exit
$ ls -l bin/CRA*
-rwxr-xr-t  2 user      group          71 Jul  8 12:31 bin/CRASTART
```

注: **chmod** コマンドを使用するには、少なくとも、セキュリティー・ソフトウェアの UNIXPRIV クラス内の SUPERUSER.FILESYS.CHANGEPERMS プロファイルに対する READ アクセス権が必要です。あるいは、このプロファイルが定義されていない場合は、スーパーユーザー (UID 0) であることが必要です。詳細については、「*UNIX System Services 計画*」(GA88-8639) を参照してください。

予約済み TCP/IP ポート

netstat コマンド (TSO または z/OS UNIX) を使用すると、現在使用中のポートの概要を取得できます。このコマンドの出力は、下記の例のようになります。使用されているポートは、「Local Socket」列の最後の番号 (「..」の後) です。これらのポートは既に使用されているので、Developer for System z 構成に使用することはできません。

IPv4

MVS TCP/IP	NETSTAT	CS VxRy	TCPIP Name: TCPIP	16:36:42
User Id	Conn	Local Socket	Foreign Socket	State
-----	-----	-----	-----	-----
BPX0INIT	00000014	0.0.0.0..10007	0.0.0.0..0	Listen
INETD4	0000004D	0.0.0.0..512	0.0.0.0..0	Listen
RSED	0000004B	0.0.0.0..4035	0.0.0.0..0	Listen
JMON	00000038	0.0.0.0..6715	0.0.0.0..0	Listen

IPv6

MVS TCP/IP	NETSTAT	CS VxRy	TCPIP Name: TCPIP	12:46:25
User Id	Conn	State		
-----	-----	-----		
BPX0INIT	00000018	Listen		
	Local Socket:	0.0.0.0..10007		
	Foreign Socket:	0.0.0.0..0		
INETD4	00000046	Listen		
	Local Socket:	0.0.0.0..512		
	Foreign Socket:	0.0.0.0..0		
RSED	0000004B	Listen		
	Local Socket:	0.0.0.0..4035		
	Foreign Socket:	0.0.0.0..0		
JMON	00000037	Listen		
	Local Socket:	0.0.0.0..6715		
	Foreign Socket:	0.0.0.0..0		

もう 1 つ存在する可能性がある制限は、予約済み TCP/IP ポートです。TCP/IP ポートを予約する一般的な場所は、以下の 2 つです。

• PROFILE.TCPIP

これは TCP/IP 開始タスクの PROFILE DD ステートメントによって参照されるデータ・セットで、多くの場合、SYS1.TCPPARMS(TCPPROF) という名前が付いています。

- PORT: 指定されたジョブ名のポートを予約します。
- PORTRANGE: 指定されたジョブ名のポート範囲を予約します。

これらのステートメントの詳細については、「*Communications Server IP 構成ガイド*」(SC88-8926) を参照してください。

• SYS1.PARMLIB(BPXPRMxx)

- INADDRANYPORT: システムが PORT 0、INADDR_ANY バインドに使用するために予約するポート番号範囲の開始ポート番号を指定します。この値は、CINET (単一ホスト上でアクティブな複数の TCP/IP スタック) にのみ必要です。
- INADDRANYCOUNT: INADDRANYPORT パラメーターで指定したポート番号から始まる、システムが予約するポートの数を指定します。この値は、CINET (単一ホスト上でアクティブな複数の TCP/IP スタック) にのみ必要です。

これらのステートメントの詳細については、「*UNIX System Services 計画*」(GA88-8639) および「*MVS 初期設定およびチューニング解説書*」(SA88-8564) を参照してください。

これらの予約済みポートは、**netstat portl** コマンド (TSO または z/OS UNIX) でリストできます。このコマンドは、以下の例のような出力を作成します。

MVS TCP/IP NETSTAT CS VxRy	TCPIP Name: TCPIP	17:08:32
Port# Prot User Flags Range IP Address		
00007 TCP MISC SERV DA		
00009 TCP MISC SERV DA		
00019 TCP MISC SERV DA		
00020 TCP OMVS D		
00021 TCP FTPD1 DA		
00025 TCP SMTP DA		
00053 TCP NAMESRV DA		
00080 TCP OMVS DA		
03500 TCP OMVS DAR	03500-03519	
03501 TCP OMVS DAR	03500-03519	

NETSTAT コマンドの詳細については、「*Communications Server IP* システム管理者のコマンド」(SC88-9073) を参照してください。

注: **NETSTAT** コマンドは PROFILE.TCPIP 内で定義された情報のみを表示します。これは、BPXPRMxx 定義とオーバーラップします。疑義または問題がある場合は、BPXPRMxx parmlib メンバーを検査して、ここで予約されるポートを確認してください。

アドレス・スペース・サイズ

RSE デーモンは z/OS UNIX Java プロセスであり、機能を実行するために大きな領域サイズを必要とします。したがって、OMVS アドレス・スペースに大きなストレージ限度を設定することが重要です。

始動 JCL の要件

RSE デーモンは、BPXBATSL (この領域サイズは 0 であることが必要です) を使用して JCL によって始動されます。

SYS1.PARMLIB(BPXPRMxx) で設定される制限

SYS1.PARMLIB(BPXPRMxx) で MAXASSIZE を設定してください。これは、デフォルトの OMVS アドレス・スペース (プロセス) 領域サイズを 2G に定義します。これは、許容される最大サイズです。これはシステム全体の限度であるため、すべての z/OS UNIX アドレス・スペースに対してアクティブとなります。これが望ましい値でない場合は、ご使用のセキュリティー・ソフトウェアで Developer for System z 独自の限度を設定できます。

この値は、「MVS システム・コマンド」(GC88-6592) で説明されているように、以下のコンソール・コマンドで検査し、動的に (次回の IPL まで) 設定できます。

1. DISPLAY OMVS,0
2. SETOMVS MAXASSIZE=2G

セキュリティー・プロファイル内に保管される制限

デーモンのユーザー ID OMVS セグメント内で ASSIZEMAX を検査し、2147483647 に設定するか、できれば NONE に設定して SYS1.PARMLIB(BPXPRMxx) 値を使用してください。

RACF を使用している場合は、「*Security Server RACF コマンド言語解説書*」(SA88-8617) で説明されているように、以下の TSO コマンドでこの値を検査および設定できます。

1. LISTUSER userid NORACF OMVS
2. ALTUSER userid OMVS(NOASSIZEMAX)

システム出口によって強制される制限

必ず、システム出口 IEFUSI または IEALIMIT が OMVS アドレス・スペース領域サイズを制御できないようにしてください。これを実現する 1 つの方法は、SYS1.PARMLIB(SMFPRMxx) 内に SUBSYS(OMVS,NOEXITS) をコーディングすることです。

SYS1.PARMLIB(SMFPRMxx) 値は、「*MVS システム・コマンド*」(GC88-6592) で説明されているように、以下のコンソール・コマンドで検査およびアクティブ化することができます。

1. DISPLAY SMF,0
2. SET SMF=xx

64 ビット・アドレッシングでの制限

SYS1.PARMLIB(SMFPRMxx) 内の MEMLIMIT キーワードでは、64 ビット・タスクが 2 GB 境界より上に割り振ることのできる仮想ストレージの量を制限します。JCL の REGION パラメーターとは異なり、MEMLIMIT=0M は、プロセスが 2 GB 境界より上の仮想ストレージを使用できないことを意味します。

SMFPRMxx に MEMLIMIT を指定しない場合は、デフォルト値 0M が使用されるため、タスクは (31 ビット) 2 GB 境界より下の 2 GB にバインドされます。このデフォルトは z/OS 1.10 で 2G に変更されて、64 ビット・タスクが 4 GB まで使用できるようになりました (2 GB 境界より下の 2 GB および MEMLIMIT で許可される 2 GB 境界より上の 2 GB)。

SYS1.PARMLIB(SMFPRMxx) 値は、「*MVS システム・コマンド*」(GC88-6592) で説明されているように、以下のコンソール・コマンドで検査およびアクティブ化することができます。

1. DISPLAY SMF,0
2. SET SMF=xx

MEMLIMIT は、JCL で EXEC カードのパラメーターとして指定することもできます。MEMLIMIT パラメーターを指定しない場合は、SMF に定義された値がデフォルトとなります。ただし、REGION=0M を指定した場合は、デフォルトが NOLIMIT となります。

各種情報

エラー・フィードバック B37 スペース異常終了

ユーザーがコンパイル・アクション時にエラー・フィードバックを選択した場合、Developer for System z によっていくつかの一時データ・セットが作成されます。こ

これらのデータ・セットのいずれかがスペース不足になった場合に、コンパイル・ジョブは B37-04 スペース異常終了で終了します。

ユーザーがこの問題を経験した場合は、FEK.SFEKPROC(FEKFERRF) 内のスペース割り振りを調整してください。デフォルト値は SPACE(200,40) TRACKS です。

システム限度

SYS1.PARMLIB(BPXPRMxx) には、複数の Developer for System z クライアントがアクティブなときに到達する可能性がある、z/OS UNIX に関連する多数の限度が定義されています。大部分の BPXPRMxx 値は、**SETOMVS** および **SET OMVS** コンソール・コマンドで動的に変更できます。

BPXPRMxx のいずれかの限度に到達しそうなときに z/OS UNIX でコンソール・メッセージ (BPXI040I) を表示させるには、**SETOMVS LIMMSG=ALL** コンソール・コマンドを使用します。

接続の拒否

各 RSE 接続ごとに複数のプロセスが開始され、それらは永続的にアクティブになります。新規接続は、特にユーザーが同じ UID を共用している場合 (デフォルトの OMVS セグメントを使用している場合など) には、プロセスの量について SYS1.PARMLIB(BPXPRMxx) で設定された限度のために、拒否されることがあります。

- 1 つの UID 当たりの限度は MAXPROCUSER キーワードによって設定され、デフォルト値は 25 です。
- システム全体の限度は MAXPROCSYS キーワードによって設定され、デフォルト値は 200 です。

接続が拒否されるもう 1 つの原因は、アクティブな z/OS アドレス・スペースと z/OS UNIX ユーザー数に課された限度です。

- アドレス・スペース ID (ASID) の最大数は SYS1.PARMLIB(IEASYSxx) 内で MAXUSER キーワードによって定義され、デフォルト値は 255 です。
- z/OS UNIX ユーザー ID (UID) の最大数は SYS1.PARMLIB(BPXPRMxx) 内の MAXUIDS キーワードによって定義され、デフォルト値は 200 です。

OutOfMemoryError

RSE スレッド・プールは、OutOfMemoryError メッセージをログに記録して失敗することがあります。このエラーは Java ヒープ・サイズに関連しており、このスレッド・プールでアクティブなユーザーが予期されたより多くのリソースを使用している場合に発生することがあります。このエラーの一般的な原因は、次のとおりです。

- リモート・システム・エクスプローラーで大きいデータ・セット・フィルターを展開した。
- 大量のメンバーが含まれる PDS(E) をオープンした。
- 大量のメンバーまたは順次ファイルをオープンした。

この問題を解決するための処置は、次のとおりです。

- `rsed.envvars` 内で `-Xmx` ディレクティブの値を大きくしてください。このディレクティブは、Java ヒープの最大サイズを制御します。Java ヒープは、アドレス・スペース制限の範囲内でなければならないことに注意してください。
- `rsed.envvars` 内で `-Dmaximum.clients` ディレクティブの値を小さくしてください。このディレクティブは、1 つのスレッド・プールに配置できるユーザー数 (これらのユーザーは 1 つの Java ヒープを共用する) を制御します。

Host Connect Emulator

- Host Connect Emulator は、ホストへの接続に RSE サーバーでなく、TN3270 Telnet を使用します。
- セキュア Telnet (SSL) を使用し、既知の CA による署名がない証明書を使用して作業している場合、すべてのクライアントは、トラステッド CA の Host Connect Emulator リストに CA 証明書を追加する必要があります。
- SNA 機能拡張を使用不可にするために、TCP/IP の TELNETPARMS の NOSNAEXT オプションが必要になる場合があります。NOSNAEXT を指定した場合、TN3270 Telnet サーバーは競合の解決と SNA センス機能についてネゴシエーションを行いません。

第 13 章 SSL および X.509 認証のセットアップ

この付録は、Secure Socket Layer (SSL) のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。また、この付録には、X.509 証明書で自分自身を認証するユーザーをサポートするためのサンプルのセットアップも記載されています。

セキュアな通信は、正しい通信パートナーと通信できるようにし、他の人間がデータを傍受して読み取ることが困難な方法で情報を伝送することを意味します。SSL は、TCP/IP ネットワーク内でその機能を提供します。SSL では、デジタル証明書を使用して自分自身を識別し、公開鍵プロトコルを使用して通信を暗号化します。SSL で使用されるデジタル証明書および公開鍵プロトコルの詳細については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

Developer for System z 用に SSL 通信をセットアップするために必要なアクションは、厳密な必要性、使用される RSE 通信方式、およびサイトで既に使用可能なものに応じてサイトごとに異なります。

この付録では、現行の RSE 定義のクローンを作成して、SSL を使用する 2 番目の RSE デーモン接続を使用できるようにします。また、RSE 接続のさまざまな部分で使用される独自のセキュリティ証明書も作成します。

- 206 ページの『暗号化方式として SSL または TLS のどちらを使用するかを決定する』
- 206 ページの『秘密鍵と証明書を保管する場所の決定』
- 207 ページの『RACF による鍵リングの作成』
- 209 ページの『既存の RSE セットアップのクローン作成』
- 210 ページの『共存を可能にするための rsed.envvars の更新』
- 210 ページの『SSL を有効にするための ssl.properties の更新』
- 211 ページの『新しい RSE デーモンの作成による SSL のアクティブ化』
- 211 ページの『接続のテスト』
- 214 ページの『(オプション) X.509 クライアント認証サポートの追加』
- 215 ページの『(オプション) gskkyman による鍵データベースの作成』
- 217 ページの『(オプション) keytool による鍵ストアの作成』

この付録では、次のような一律の命名規則が使用されています。

- 証明書: rdzrse
- 鍵および証明書ストレージ: rdzssl.*
- パスワード: rsessl
- デーモン・ユーザー ID: stcrse

以下に述べるタスクの一部では、ユーザーが z/OS UNIX 内でアクティブであることを想定しています。そのためには、TSO コマンド **OMVS** を発行します。TSO に戻るには、**exit** コマンドを使用します。

暗号化方式として SSL または TLS のどちらを使用するかを決定する

`rsed.envvars` の `_RSE_JAVA_OPTS` ディレクティブの `DSTORE_SSL_ALGORITHM` 変数によって、SSL とその後継の TLS (トランスポート層セキュリティ) を暗号化方式として選択できます。詳しくは、「ホスト構成ガイド」(SC88-5663) の『`_RSE_JAVA_OPTS` での追加 Java 始動パラメーターの定義』を参照してください。

秘密鍵と証明書を保管する場所の決定

SSL で使用される ID 証明書と暗号化/暗号化解除鍵は、鍵ファイルに保管されます。この鍵ファイルには、アプリケーション・タイプに応じて、さまざまな実装が存在します。

しかし、すべての実装が同じ原則に従います。あるコマンドが、鍵ペア (公開鍵とそれに関連付けられた秘密鍵) を生成します。その後、コマンドは公開鍵を X.509 自己署名証明書の中に包み込み、この証明書は単一要素の証明書チェーンとして保管されます。この証明書チェーンと秘密鍵が (別名によって識別される) 1 つの項目として、鍵ファイルに保管されます。

RSE デーモンは System SSL アプリケーションであり、鍵データベース・ファイルを使用します。この鍵データベースは、`gskkyman` によって作成された物理ファイルとするか、SAF 準拠のセキュリティ・ソフトウェア (例えば RACF) などによって管理される鍵リングとすることができます。RSE サーバー (これはデーモンによって始動されます) は Java SSL アプリケーションであり、`keytool` によって作成される鍵ストア・ファイルを使用するか、セキュリティ・ソフトウェアによって管理される鍵リングを使用します。

表 39. SSL 証明書の保管メカニズム

証明書ストレージ	作成者および管理者	RSE デーモン	RSE サーバー
鍵リング	SAF 準拠のセキュリティ製品	サポート	サポート
鍵データベース	z/OS UNIX の <code>gskkyman</code>	サポート	/
鍵ストア	Java の <code>keytool</code>	/	サポート

SSL を通じて接続するためには、鍵ストアと鍵データベースの両方が、z/OS UNIX ファイルとして、または SAF 準拠の鍵リングとして必要です。

- 鍵ストア (RACF または `keytool`)
- 鍵データベース (RACF または `gskkyman`)

注:

- SAF 準拠の鍵リングは、証明書の管理に推奨される方式です。
- RSE デーモンと RSE サーバーが同じ証明書管理方式を使用する場合は、共用証明書を使用できます。

- RSE デーモンは、プログラム制御で実行する必要があります。内部で System SSL を使用することは、SYS1.SIEALNKE が、ご使用のセキュリティー・ソフトウェアによってプログラム制御されている必要があることを意味します。
- System SSL アプリケーション (デーモン接続) を実行するためには、SYS1.SIEALNKE が LINKLIST または STEPLIB の中に存在する必要があります。STEPLIB 方式を使用したい場合は、次のステートメントを rsed.envvars の末尾に追加します。

```
STEPLIB=$STEPLIB:SYS1.SIEALNKE
```

ただし、以下の点に注意してください。

- STEPLIB を z/OS UNIX で使用すると、パフォーマンスに悪い影響が出ます。
- 1 つの STEPLIB ライブラリーに APF 許可がある場合、すべての STEPLIB ライブラリーに APF 許可があることが必要です。ライブラリーは、STEPLIB 内で許可のないライブラリーと混用した場合、APF 許可を失います。
- System SSL は、Integrated Cryptographic Service Facility (ICSF) を使用します (使用可能な場合)。ICSF は、System SSL ソフトウェア・アルゴリズムの代わりに使用されるハードウェア暗号化サポートを提供します。詳細については、「System SSL プログラミング」(SD88-6252) を参照してください。

RACF およびデジタル証明書については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。gskkyman の資料は、「System SSL プログラミング」(SD88-6252) で、keytool の資料は、<http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html> で入手できます。

RACF による鍵リングの作成

RSE デーモン鍵データベースの作成に gskkyman を使用し、RSE サーバー鍵ストアの作成に keytool を使用する場合は、このステップを実行しないでください。

RACDCERT コマンドは、秘密鍵と証明書を RACF にインストールし、保守します。RACF は、複数の秘密鍵と証明書を 1 つのグループとして管理できます。それらのグループは、鍵リングと呼ばれます。

RACDCERT コマンドの詳細については、「*Security Server RACF コマンド言語解説書*」(SA88-8617) を参照してください。

```
RDEFINE FACILITY IRR.DIGTCERT.LIST UACC(NONE)
RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY) ACCESS(READ) ID(stcrse)
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(READ) ID(stcrse)
SETROPTS RACLIST(FACILITY) REFRESH
```

```
RACDCERT ID(stcrse) GENCERT SUBJECTSDN(CN('rdz rse ssl') +
OU('rdz') O('IBM') L('Raleigh') SP('NC') C('US')) +
NOTAFTER(DATE(2017-05-21)) WITHLABEL('rdzrse') KEYUSAGE(HANDSHAKE)
```

```
RACDCERT ID(stcrse) ADDRING(rdzssl.racf)
RACDCERT ID(stcrse) CONNECT(LABEL('rdzrse') RING(rdzssl.racf) +
DEFAULT USAGE(PERSONAL))
```

上記のサンプルは始めに、必要なプロファイルを作成し、ユーザー ID STCRSE に、そのユーザー ID が所有する証明書と鍵リングへのアクセスを許可します。使用するユーザー ID は、SSL RSE デーモンを実行するために使用したユーザー ID に一

致する必要があります。次のステップは、新しい自己署名証明書を `rdzrse` というラベルで作成することです。パスワードは必要ありません。この証明書は、その後、新規に作成された鍵リング (`rdzssl.racf`) に追加されます。証明書の場合と同様に、鍵リングにパスワードは必要ありません。

結果は、以下の `list` オプションを使用して検査できます。

```
RACDCERT ID(stcrse) LIST
Digital certificate information for user STCRSE:

Label: rdzrse
Certificate ID: 2QjW10Xi0sXZ1aaEqZmihUBA
Status: TRUST
Start Date: 2007/05/24 00:00:00
End Date: 2017/05/21 23:59:59
Serial Number:
>00<
Issuer's Name:
>CN=rdz rse ssl.OU=rdz.O=IBM.L=Raleigh.SP=NC.C=US<
Subject's Name:
>CN=rdz rse ssl.OU=rdz.O=IBM.L=Raleigh.SP=NC.C=US<
Private Key Type: Non-ICSF
Private Key Size: 1024
Ring Associations:
Ring Owner: STCRSE
Ring:
>rdzssl.racf<
```

(オプション) 署名付き証明書の使用

証明書には、自己署名付きの証明書と認証局 (CA) によって署名された証明書があります。CA によって署名された証明書は、その証明書の所有者が本人に間違いのないことを CA が保証することを意味します。署名プロセスでは、CA 資格情報 (証明書でもある) がユーザーの証明書に追加され、複数要素の証明書チェーンが作成されます。

CA によって署名された証明書を使用した場合は、Developer for System z クライアントによる信頼性検証のための質問を回避できます (クライアントがすでにその CA を信頼している場合)。

CA 署名付き証明書を作成および使用するには、以下のステップを実行します。

1. 自己署名証明書を作成します。
`RACDCERT ID(stcrse) GENCERT WITHLABEL('rdzrse') . . .`
2. この証明書についての署名要求を作成します。
`RACDCERT ID(stcrse) GENREQ (LABEL('rdzrse')) DSN(dsn)`
3. 署名要求を、選択した CA に送信します。
4. CA 資格情報 (証明書でもある) が既知であるかどうかを検査します。
`RACDCERT CERTAUTH LIST`
5. CA 証明書に、信頼できるものとしてのマークを付けます。
`RACDCERT CERTAUTH ALTER(LABEL('CA cert')) TRUST`

または、CA 証明書をデータベースに追加します。
`RACDCERT CERTAUTH ADD(dsn) WITHLABEL('CA cert') TRUST`

- 署名付き証明書をデータベースに追加します。これにより、自己署名証明書が置き換えられます。

```
RACDCERT ID(stcrse) ADD(dsn) WTIHLABEL('rdzrse') TRUST
```

注: 自己署名証明書を、置き換える前に削除しないでください。削除すると、証明書に付随する秘密鍵が失われ、証明書が役に立たなくなります。

- 鍵リングを作成します。

```
RACDCERT ID(stcrse) ADDRING(rdzssl.racf)
```

- 署名付き証明書を鍵リングに追加します。

```
RACDCERT ID(stcrse) CONNECT(ID(stcrse) LABEL('rdzrse')  
RING(rdzssl.racf))
```

- CA 証明書を鍵リングに追加します。

```
RACDCERT ID(stcrse) CONNECT(CERTAUTH LABEL('CA cert')  
RING(rdzssl.racf))
```

証明書の署名に使用する CA 証明書は、同様に、別の高位の CA 証明書により署名されることもあるため、注意が必要です。その場合、当該の高位の CA 証明書も、鍵リングに追加する必要があります。このプロセスは、高位の CA 証明書がルート of CA 証明書になるまで繰り返します。ルートの CA 証明書は、必ず自己署名証明書になります。

既存の RSE セットアップのクローン作成

このステップでは、SSL セットアップを既存のセットアップと並行して稼働できるように、RSE 構成ファイルの新しいインスタンスを作成します。以下のサンプル・コマンドでは、構成ファイルが /etc/rdz/ に入っていることを想定しています。これは、「ホスト構成ガイド」(SC88-5663)の『カスタマイズ・セットアップ』"で使われるデフォルトのロケーションです。

```
$ cd /etc/rdz  
$ mkdir ssl  
$ cp rsed.envvars ssl  
$ cp ssl.properties ssl  
$ ls ssl  
rsed.envvars    ssl.properties
```

上記の例の z/OS UNIX コマンドは ssl というサブディレクトリーを作成し、そのサブディレクトリーに変更が必要な構成ファイルを取り込みます。その他の構成ファイル、インストール・ディレクトリー、および MVS コンポーネントは、SSL 固有ではないので、共用できます。

既存の大部分の構成ファイルを再利用することにより、SSL のセットアップに実際に必要な変更集中でき、RSE セットアップ全体を再度実行しなくても済みます。(例えば、ISPF.conf の新しいロケーションを定義せずに済みます。)

共存を可能にするための rsed.envvars の更新

これまでのところ、定義は現行のセットアップの正確なコピーであり、これは、新しい RSE デーモンのログが現行のサーバー・ログ・ファイルと重なることを意味します。また、RSE は、ssl ディレクトリーにコピーされなかった構成ファイルがどこにあるかも知っている必要があります。どちらの問題も、rsed.envvars に小さな変更を加えることによって対処できます。

```
$ oedit /etc/rdz/ssl/rsed.envvars
-> change: _RSE_RSED_PORT=4047
-> change: -Ddaemon.log=/var/rdz/logs/ssl
-> change: -Duser.log=/var/rdz/logs/ssl
-> add at the END:
# -- NEEDED TO FIND THE REMAINING CONFIGURATION FILES
CFG_BASE=/etc/rdz
CLASSPATH=.:$CFG_BASE:$CLASSPATH
# --
```

上記の例の変更では、新しいログ・ロケーションが定義されます (このログ・ロケーションが存在しない場合は、RSE デーモンによって作成されます)。また、これらの変更では、SSL RSE プロセスが最初に現行ディレクトリー (/etc/rdz/ssl) で構成ファイルを検索し、その後に元のディレクトリー (/etc/rdz) を検索するように CLASSPATH も更新されます。

SSL を有効にするための ssl.properties の更新

ssl.properties を更新することにより、SSL 暗号化通信を使用して開始するように RSE に指示します。

```
$ oedit /etc/rdz/ssl/ssl.properties
-> change: enable_ssl=true
-> uncomment and change: daemon_keydb_file=rdzssl.racf
-> uncomment and change: daemon_key_label=rdzrse
-> uncomment and change: server_keystore_file=rdzssl.racf
-> uncomment and change: server_keystore_label=rdzrse
-> uncomment and change: server_keystore_type=JCERACFKS
```

上記の例の変更では SSL を使用可能に設定してから、RSE デーモンおよび RSE サーバーに対して、(共有の) 証明書が鍵リング rdzssl.racf 内のラベル rdzrse の下に保管されていることを通知します。JCERACFKS キーワードは、SAF 準拠の鍵リングが鍵ストアとして使用されることを RSE サーバーに通知します。

デーモンが使用する System SSL は、常に ICSF を使用することに注意してください (使用可能な場合)。この ICSF は System z 暗号化ハードウェアへのインターフェースです。ICSF 使用時にデーモン定義をサーバーと共有できるようにするには、server_keystore_type JCECCARACFKS を指定する必要があります。ここでは、SAF 準拠の鍵リングが公開鍵の鍵ストアとしても使用されていますが、秘密鍵は ICSF に保管されます。「*Cryptographic Services ICSF Administrator's Guide*」(SA22-7521) で説明されているように、ICSF は CSFKEYS および CSFSERV セキュリティー・クラス内のプロファイルを使用して、暗号鍵と暗号サービスを使用できるユーザーを制御します。

新しい RSE デーモンの作成による SSL のアクティブ化

前に述べたように、ここでは SSL を使用する第 2 の接続を作成します。これは、新しい RSE デーモンを作成することを意味します。RSE デーモンは、開始タスクまたはユーザー・ジョブとすることができます。ここでは、初期 (テスト) セットアップにユーザー・ジョブ方式を使用します。以下の説明では、サンプル JCL が FEK.#CUST.PROCLIB(RSED) に入っていることを想定しています。これは、「ホスト構成ガイド」(SC88-5663)の"『カスタマイズ・セットアップ』"で使用されるデフォルトのロケーションです。

1. 新しいメンバー FEK.#CUST.PROCLIB(RSEDSSL) を作成し、サンプル JCL FEK.#CUST.PROCLIB(RSED) の中にコピーします。
2. 先頭にジョブ・カードを、末尾に exec ステートメントをそれぞれ追加して、RSEDSSL をカスタマイズします。また、以下のコード・サンプルに示すように、SSL に関連する構成ファイル (/etc/rdz/ssl) のロケーションも指定します。ここでは、ユーザー ID STCRSE を使用する点に注意してください。このユーザー ID には、前のステップで、証明書と鍵リングに対する正しいアクセス権限が与えられているためです。

```
//RSEDSSL JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1),USER=STCRSE
/*
/* RSE DAEMON - SSL
/*
//RSED      PROC TMPDIR=,
//          PORT=,
//          IVP=,                * 'IVP' to do an IVP test
//          CNFG='/etc/rdz/ssl',
//          HOME='/usr/lpp/rdz'
/*
//RSED      EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,
// PARM='PGM &HOME./bin/rsed.sh &IVP -C&CNFG -P&PORT -T&TMPDIR'
//STDOUT   DD SYSOUT=*
//STDERR    DD SYSOUT=*
//          PEND
/*
//RSED      EXEC RSED
/*
```

図 30. RSEDSSL - SSL 用の RSE デーモン・ユーザー・ジョブ

注: RSEDSSL ジョブに割り当てられるユーザー ID は、元の RSE デーモンと同じ権限を持っている必要があります。FACILITY プロファイル BPX.SERVER と PTKTDATA プロファイル IRRPTAUTH.FEKAPPL.* が、ここでの重要な要素です。

接続のテスト

SSL ホスト構成が完了し、前に作成したジョブ FEK.#CUST.PROCLIB(RSEDSSL) を実行依頼することにより、SSL 用の RSE デーモンを始動できます。

この時点で、Developer for System z クライアントと接続することにより、新しいセットアップをテストできます。SSL で使用するために新しい構成を (既存の構成のクローン作成によって) 作成してあるので、RSE デーモン用のポート 4047 を使用して、クライアント上に新しい接続を定義する必要があります。

接続と同時に、ホストとクライアントはセキュア・パスをセットアップするために何らかのハンドシェークを開始します。このハンドシェークの一環として、証明書の交換があります。Developer for System z クライアントは、ホスト証明書またはそれに署名した CA を認識できない場合、ユーザーにその証明書が信頼できるかどうかを尋ねるプロンプトを出します。

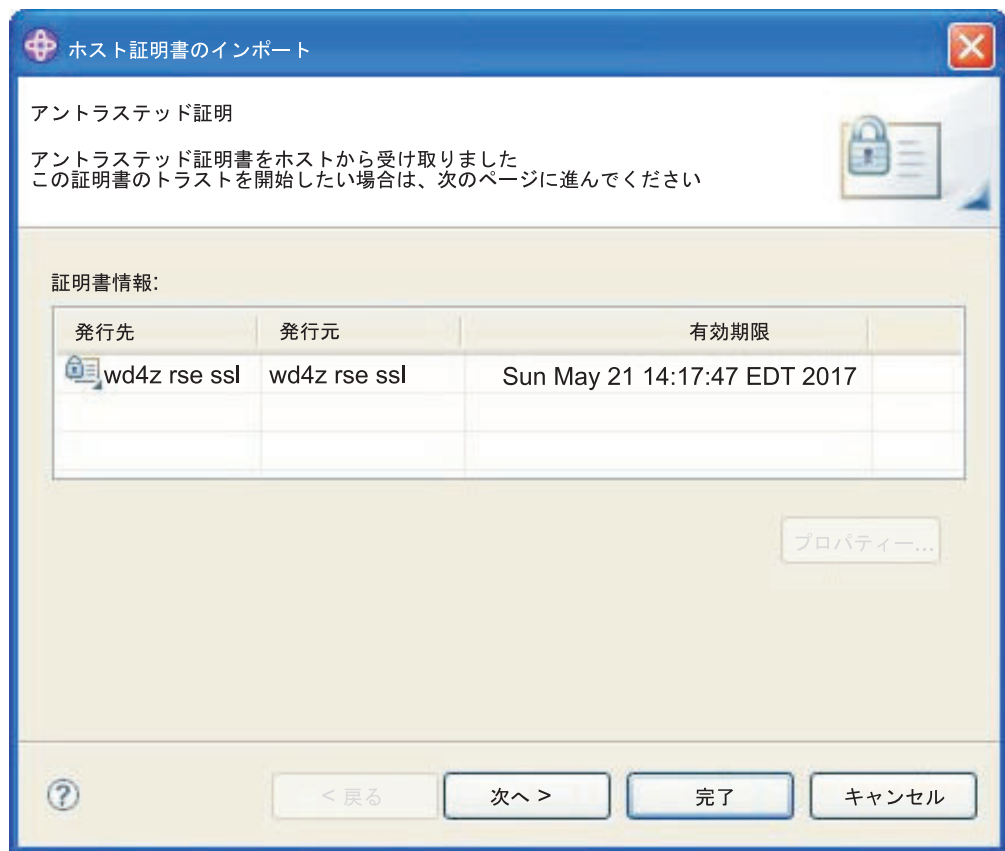


図 31. 「ホスト証明書のインポート」ダイアログ

「完了」ボタンをクリックすると、この証明書を信頼できるものとして受け入れることができ、その後、接続の初期化が続行されます。

注: RSE デーモンと RSE サーバーは 2 つの異なる証明書ロケーションを使用する場合があります、その結果、2 つの異なる証明書が使用され、したがって 2 つの確認が行われる場合があります。

証明書がクライアントに既知となった後、このダイアログが再び表示されることはありません。信頼できる証明書のリストは、「ウィンドウ」>「設定...」>「リモート・システム」>「SSL」を選択することによって管理できます。その場合、次のダイアログが表示されます。

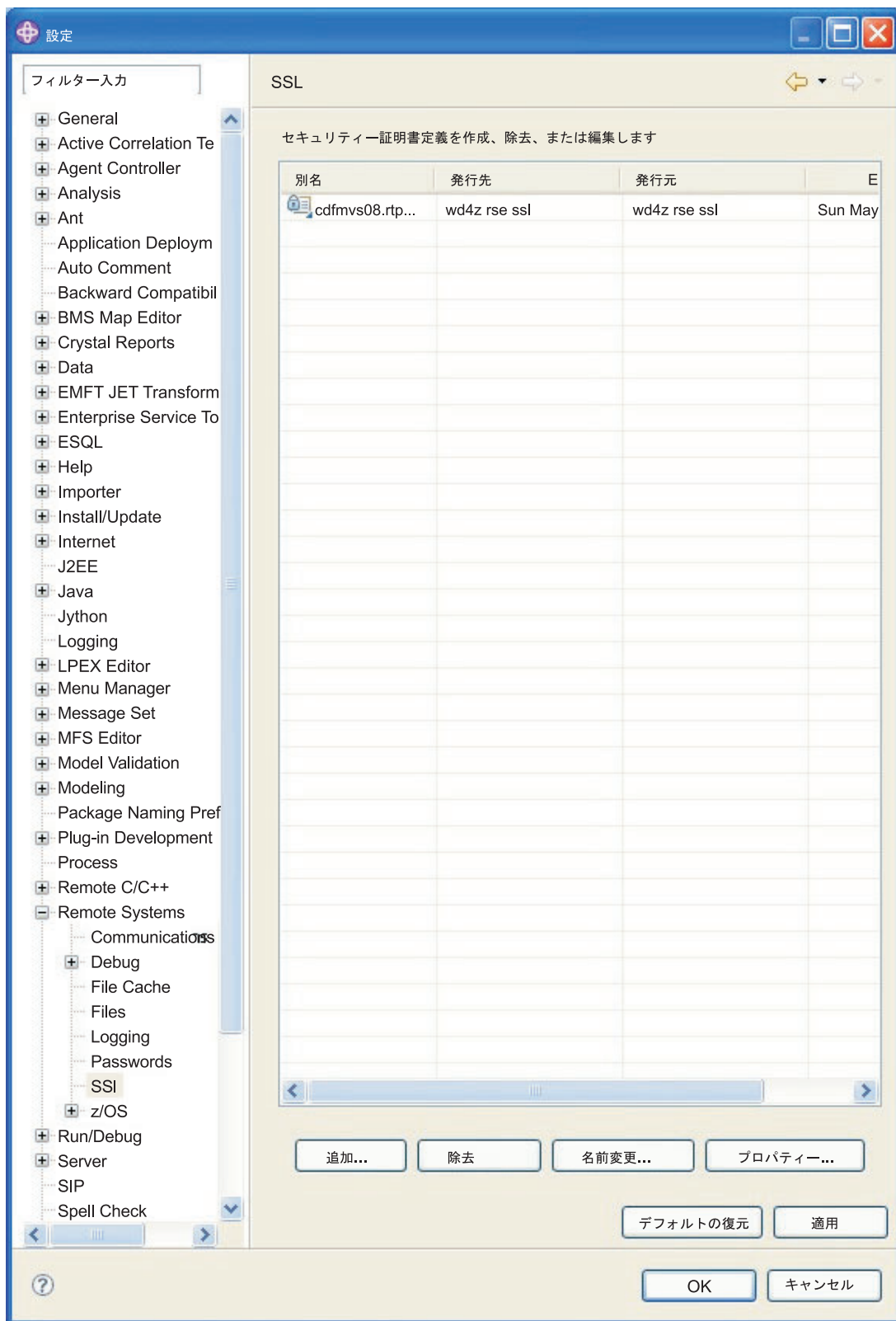


図 32. 「設定」 ダイアログ - 「SSL」

SSL 通信が失敗した場合、クライアントはエラー・メッセージを返します。詳細な情報は、186 ページの『RSE デーモンおよびスレッド・プールのロギング』、およ

び 187 ページの『RSE ユーザー・ロギング』の説明のように、さまざまなサーバーおよびユーザー・ログ・ファイルから入手できます。

(オプション) X.509 クライアント認証サポートの追加

RSE デーモンは、X.509 証明書で自分自身を認証するユーザーをサポートしています。この機能の場合、SSL 暗号化通信の使用は前提条件です。この機能は、SSL で使用される証明書でのホスト認証を拡張したものです。

ユーザーの証明書認証を行うには、35 ページの『クライアント認証、X.509 証明書を使用した』の説明にあるように、複数の方法があります。以下の手順は、セキュリティー・ソフトウェアで HostIdMappings 証明書拡張を使用して証明書を認証する方式をサポートするために必要なセットアップを文書化したものです。

1. クライアント証明書への署名に使用された認証局 (CA) を識別する証明書を、高度に信頼できる CA 証明書に変更します。証明書の妥当性検査には TRUST 状況で十分ですが、ログオン・プロセスの証明書認証の部分で HIGHTRUST が使用されるので、HIGHTRUST に変更します。

```
RACDCERT CERTAUTH ALTER(LABEL('HighTrust CA')) HIGHTRUST
```

2. CA 証明書を鍵リング rdzssl.racf に追加します。これにより、CA 証明書をクライアント証明書の妥当性検査に使用できるようになります。

```
RACDCERT ID(stcrse) CONNECT(CERTAUTH LABEL('HighTrust CA') +  
RING(rdzssl.racf))
```

これで、CA 証明書のためのセキュリティー・ソフトウェアのセットアップは完了しました。

3. クライアント証明書の HostIdMappings 拡張で定義されているホスト名 CDFMVS08.RALEIGH.IBM.COM 用に、SERVAUTH クラス内にリソースを定義します (フォーマット IRR.HOST.hostname)。

```
RDEFINE SERVAUTH IRR.HOST.CDFMVS08.RALEIGH.IBM.COM UACC(NONE)
```

4. RSE 開始タスク・ユーザー ID STCRSE に、このリソースに対する READ アクセス権限を付与します。

```
PERMIT IRR.HOST.CDFMVS08.RALEIGH.IBM.COM CLASS(SERVAUTH) +  
ACCESS(READ) ID(stcrse)
```

5. SERVAUTH クラスに対する変更をアクティブにします。SERVAUTH クラスがまだアクティブでない場合は、最初のコマンドを使用します。アクティブなセットアップをリフレッシュするには、2 番目のコマンドを使用します。

```
SETOPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH)  
or  
SETOPTS RACLIST(SERVAUTH) REFRESH
```

これで、HostIdMappings 拡張のためのセキュリティー・ソフトウェアのセットアップは完了しました。

6. RSE 開始タスクを再始動して、X.509 証明書を使用したクライアント・ログオンの受け入れを開始します。

(オプション) gskkyman による鍵データベースの作成

RSE デモンの鍵データベースに SAF 準拠の鍵リングを使用する場合は、このステップを実行しないでください。

gskkyman は z/OS UNIX シェル・ベースのメニュー方式プログラムであり、秘密鍵、証明書要求、および証明書が入っている z/OS UNIX ファイルを作成し、それにデータを取り込み、管理します。この z/OS UNIX ファイルは鍵データベースと呼ばれます。

注: gskkyman 用の環境をセットアップするために、以下のステートメントが必要になる場合があります。これについての詳細は、「*System SSL プログラミング*」(SD88-6252) を参照してください。

```
PATH=$PATH:/usr/lpp/gskssl/bin
export NLSPATH=/usr/lpp/gskssl/lib/nls/msg/En_US.IBM-1047/%N:$NLSPATH
export STEPLIB=$STEPLIB:SYS1.SIEALNKE
```

```
$ cd /etc/rdz/ssl
$ gskkyman          Database Menu
```

1 - Create new database

```
Enter option number: 1
Enter key database name (press ENTER to return to menu): rdzssl.kdb
Enter database password (press ENTER to return to menu): rsessl
Re-enter database password: rsessl
Enter password expiration in days (press ENTER for no expiration):
Enter database record length (press ENTER to use 2500):
```

Key database /etc/rdz/ssl/rdzssl.kdb created.

Press ENTER to continue.

Key Management Menu

6 - Create a self-signed certificate

Enter option number (press ENTER to return to previous menu): 6

Certificate Type

5 - User or server certificate with 1024-bit RSA key

```
Select certificate type (press ENTER to return to menu): 5
Enter label (press ENTER to return to menu): rdzrse
Enter subject name for certificate
  Common name (required): rdz rse ssl
  Organizational unit (optional): rdz
  Organization (required): IBM
  City/Locality (optional): Raleigh
  State/Province (optional): NC
  Country/Region (2 characters - required): US
Enter number of days certificate will be valid (default 365): 3650

Enter 1 to specify subject alternate names or 0 to continue: 0

Please wait .....

Certificate created.

Press ENTER to continue.
```

Key Management Menu

0 - Exit program

Enter option number (press ENTER to return to previous menu): 0

\$ ls -l rdzssl.*

total 152

-rw----- 1 IBMUSER SYS1 35080 May 24 14:24 rdzssl.kdb

-rw----- 1 IBMUSER SYS1 80 May 24 14:24 rdzssl.rdb

\$ chmod 644 rdzssl.*

\$ ls -l rdzssl.*

-rw-r--r-- 1 IBMUSER SYS1 35080 May 24 14:24 rdzssl.kdb

-rw-r--r-- 1 IBMUSER SYS1 80 May 24 14:24 rdzssl.rdb

上記のサンプルは始めに、rdzssl.kdb という鍵データベースを、パスワード rssql を使用して作成します。データベースの作成後、約 10 年間 (うるう日はカウントしません) 有効な新しい自己署名証明書を作成することにより、データベースにデータが取り込まれます。証明書は、rdzrse というラベルの下に、鍵データベースに使用されたのと同じパスワード (rssql) を使用して保管されます (これは RSE の必要条件です)。

gskkyman は、鍵データベースに (非常にセキュアな) 600 許可ビット・マスク (所有者だけがアクセスできる) を割り振ります。デーモンが鍵データベースの作成者と同じユーザー ID を使用している場合を除き、アクセス権限の設定をもっと制限の少ないものにする必要があります。chmod コマンドには、644 (所有者は読み取り/書き込み権限を持ち、全員が読み取り権限を持つ) のマスクを使用できます。

結果を検査するには、以下のように、「**Manage keys and certificates**」サブメニューで「**Show certificate information**」オプションを選択します。

\$ gskkyman

Database Menu

2 - Open database

Enter option number: 2

Enter key database name (press ENTER to return to menu): rdzssl.kdb

Enter database password (press ENTER to return to menu): rssql

Key Management Menu

1 - Manage keys and certificates

Enter option number (press ENTER to return to previous menu): 1

Key and Certificate List

1 - rdzrse

Enter label number (ENTER to return to selection menu, p for previous list): 1

Key and Certificate Menu

1 - Show certificate information

Enter option number (press ENTER to return to previous menu): 1

Certificate Information

Label: rdzrse

Record ID: 14

Issuer Record ID: 14

```

        Trusted: Yes
        Version: 3
Serial number: 45356379000ac997
    Issuer name: rdz rse ssl
                rdz
                IBM
                Raleigh
                NC
                US
    Subject name: rdz rse ssl
                rdz
                IBM
                Raleigh
                NC
                US
    Effective date: 2007/05/24
    Expiration date: 2017/05/21
Public key algorithm: rsaEncryption
    Public key size: 1024
Signature algorithm: sha1WithRsaEncryption
    Issuer unique ID: None
    Subject unique ID: None
Number of extensions: 3

```

Enter 1 to display extensions, 0 to return to menu: 0

Key and Certificate Menu

0 - Exit program

Enter option number (press ENTER to return to previous menu): 0

次の ssl.properties のサンプルは、daemon_* ディレクティブが、前に示した SAF 鍵リングのサンプルと異なることを示しています。

```

$ oedit /etc/rdz/ssl/ssl.properties
-> change: enable_ssl=true
-> uncomment and change: daemon_keydb_file=rdzssl.kdb
-> uncomment and change: daemon_keydb_password=rsessl
-> uncomment and change: daemon_key_label=rdzrse
-> uncomment and change: server_keystore_file=rdzssl.racf
-> uncomment and change: server_keystore_label=rdzrse
-> uncomment and change: server_keystore_type=JCERACFKS

```

上記の変更は SSL を使用可能に設定し、RSE デーモンに、パスワード rsessl を持つ鍵データベース rdzssl.kdb 内のラベル rdzrse の下に証明書が保管されることを通知します。RSE サーバーは、依然として SAF 準拠の鍵リングを使用しています。

(オプション) keytool による鍵ストアの作成

RSE サーバーの鍵ストアに SAF 準拠の鍵リングを使用する場合は、このステップを実行しないでください。

「keytool -genkey」は、秘密鍵ペアとそれに適合する自己署名証明書を生成します。それらは、(別名によって識別される) 1 つの項目として (新しい) 鍵ストア・ファイルに保管されます。

注: コマンド検索ディレクトリーに Java が含まれている必要があります。keytool を実行するには、以下のステートメントが必要になる場合があります。ここで、/usr/lpp/java/J5.0 は Java がインストールされているディレクトリーです。PATH=\$PATH:/usr/lpp/java/J5.0/bin

すべての情報を 1 つのパラメーターとして渡すことができますが、コマンド行の長さに制限があるため、以下のように何らかの対話性が必要になります。

```
$ cd /etc/rdz/ssl
$ keytool -genkey -alias rdzrse -validity 3650 -keystore rdzssl.jks -storepass
  rsessl -keypass rsessl
What is your first and last name?
  [Unknown]: rdz rse ssl
What is the name of your organizational unit?
  [Unknown]: rdz
What is the name of your organization?
  [Unknown]: IBM
What is the name of your City or Locality?
  [Unknown]: Raleigh
What is the name of your State or Province?
  [Unknown]: NC
What is the two-letter country code for this unit?
  [Unknown]: US
Is CN=rdz rse ssl, OU=rdz, O=IBM, L=Raleigh, ST=NC, C=US correct? (type "yes"
or "no")
  [no]: yes
$ ls -l rdzssl.*
-rw-r--r--  1 IBMUSER  SYS1          1224 May 24 14:17 rdzssl.jks
```

上記の例で作成した自己署名証明書は、約 10 年間有効です (うるう日はカウントしません)。これは、別名 rdzrse を使用して /etc/rdz/ssl/rdzssl.jks に保管されます。そのパスワード (rsessl) は鍵ストア・パスワードと同一であり、これは RSE の必要条件です。

結果は、以下のように -list オプションを使用して検査できます。

```
$ keytool -list -alias rdzrse -keystore rdzssl.jks -storepass rsessl -v
Alias name: rdzrse
Creation date: May 24, 2007
Entry type: keyEntry
Certificate chain length: 1
Certificate 1:
Owner: CN=rdz rse ssl, OU=rdz, O=IBM, L=Raleigh, ST=NC, C=US
Issuer: CN=rdz rse ssl, OU=rdz, O=IBM, L=Raleigh, ST=NC, C=US
Serial number: 46562b2b
Valid from: 5/24/07 2:17 PM until: 5/21/17 2:17 PM
Certificate fingerprints:
    MD5:  9D:6D:F1:97:1E:AD:5D:B1:F7:14:16:4D:9B:1D:28:80
    SHA1: B5:E2:31:F5:B0:E8:9D:01:AD:2D:E6:82:4A:E0:B1:5E:12:CB:10:1C
```

次の ssl.properties のサンプルは、server_* ディレクティブが、前に示した SAF 鍵リングのサンプルと異なることを示しています。

```
$ oedit /etc/rdz/ssl/ssl.properties
-> change: enable_ssl=true
-> uncomment and change: daemon_keydb_file=rdzssl.racf
-> uncomment and change: daemon_key_label=rdzrse
-> uncomment and change: server_keystore_file=rdzssl.jks
-> uncomment and change: server_keystore_password=rsessl
-> uncomment and change: server_keystore_label=rdzrse
-> optionally uncomment and change: server_keystore_type=JKS
```

上記の変更は SSL を使用可能に設定し、RSE サーバーに、パスワード `rsessl` を持つ鍵ストア `rdzssl.jks` 内のラベル `rdzrse` の下に証明書が保管されることを通知します。RSE デーモンは、引き続き SAF 準拠の鍵リングを使用しています。

第 14 章 TCP/IP のセットアップ

この付録は、TCP/IP のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。

TCP/IP 構成の詳細については、「*Communications Server IP 構成ガイド*」(SC88-8926) および「*Communications Server IP 構成解説書*」(SC88-8927) を参照してください。

ホスト名依存関係

TSO コマンド・サービスに APPC を使用する場合、Developer for System z は TCP/IP が初期化時に正しいホスト名を持っているかどうか依存します。つまり、各種の TCP/IP 構成ファイルやリゾルバー構成ファイルを正しくセットアップする必要があるということです。

TCP/IP 構成は、fekfivpt インストール検査プログラム (IVP) でテストできます。このコマンドは次のサンプルのような出力を返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpt
```

```
Wed Jul 2 13:11:54 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars
```

```
-----
TCP/IP resolver configuration (z/OS UNIX search order):
-----
```

```
Resolver Trace Initialization Complete -> 2008/07/02 13:11:54.745964
```

```
res_init Resolver values:
```

```
Global Tcp/Ip Dataset = None
Default Tcp/Ip Dataset = None
Local Tcp/Ip Dataset = /etc/resolv.conf
Translation Table = Default
UserId/JobName = USERID
Caller API = LE C Sockets
Caller Mode = EBCDIC
(L) DataSetPrefix = TCPIP
(L) HostName = CDFMVS08
(L) TcpIpJobName = TCPIP
(L) DomainOrigin = RALEIGH.IBM.COM
(L) NameServer = 9.42.206.2
                  9.42.206.3
(L) NsPortAddr = 53 (L) ResolverTimeout = 10
(L) ResolveVia = UDP (L) ResolverUdpRetries = 1
(*) Options NDots = 1
(*) SockNoTestStor
(*) AlwaysWto = NO (L) MessageCase = MIXED
(*) LookUp = DNS LOCAL
```

```
res_init Succeeded
```

```
res_init Started: 2008/07/02 13:11:54.755363
```

```
res_init Ended: 2008/07/02 13:11:54.755371
```

```
*****
```

```
MVS TCP/IP NETSTAT CS V1R9 TCPIP Name: TCPIP 13:11:54
```

Tcpip started at 01:28:36 on 06/23/2008 with IPv6 enabled

host IP address:

hostName=CDFMVS08
hostAddr=9.42.112.75
bindAddr=9.42.112.75
localAddr=9.42.112.75

Success, addresses match

リゾルバーについて

リゾルバーは、プログラムに代わってネーム・サーバーにアクセスするクライアントとして機能し、名前からアドレス、またはアドレスから名前への解決を行います。要求側プログラムの照会を解決するために、リゾルバーは使用可能なネーム・サーバーにアクセスするか、ローカル定義 (例えば、`/etc/resolv.conf`、`/etc/hosts`、`/etc/ipnodes`、`HOSTS.SITEINFO`、`HOSTS.ADDRINFO`、または `ETC.IPNODES`) を使用するか、あるいはその両方の組み合わせを使用します。

リゾルバー・アドレス・スペースが開始されると、リゾルバー JCL プロシージャ内の `SETUP DD` カードによって指し示されている、オプションのリゾルバー・セットアップ・データ・セットが読み取られます。セットアップ情報が提供されていない場合、リゾルバーは該当するネイティブの MVS または z/OS UNIX 検索順序を使用します。その際、`GLOBALTCPIPDATA`、`DEFAULTTCPIPDATA`、`GLOBALIPNODES`、`DEFAULTIPNODES`、または `COMMONSEARCH` 情報は使用されません。

構成情報の検索順序について

TCP/IP 機能によって使用される構成ファイルの検索順序と、どのようなときにデフォルトの検索順序を環境変数、JCL、またはその他の変数によってオーバーライドできるかを理解しておくことが重要です。その知識があれば、ローカル・データ・セットと HFS ファイルの命名の標準に対応することができ、問題の診断時にも、使用されている構成データ・セットまたは HFS ファイルを知るのに役立ちます。

注意すべきもう 1 つの重要な点は、いずれかの構成ファイルに検索順序が適用された場合、最初に検出されたファイルで検索が終了するということです。したがって、絶対に検出されないファイル内に構成情報を入れた場合は、検索順序内に他のファイルが先に存在するか、またはアプリケーションが選択した検索順序内にそのファイルが含まれていないために、予期しない結果になることがあります。

構成ファイルを検索するときは、JCL プロシージャの中で `DD` ステートメントを使用するか環境変数を設定することにより、大部分の構成ファイルが置かれている場所を TCP/IP に明示的に指示できます。それ以外の場合は、「*Communications Server IP 構成ガイド*」(SC88-8926) で説明されている検索順序に基づいて、TCP/IP に構成ファイルのロケーションを動的に判別させることができます。

TCP/IP スタックの構成コンポーネントは、TCP/IP スタックの初期化時に `TCPIP.DATA` を使用して、スタックの `HOSTNAME` を判別します。その値を取得するために、z/OS UNIX 環境の検索順序が使用されます。

注: リゾルバーによって使用される TCPIP.DATA 値、およびそれらの値が読み取られた場所を判別するには、リゾルバー・トレース機能を使用します。トレースを動的に開始する方法については、「*Communications Server: IP Diagnosis Guide*」(GC31-8782) を参照してください。トレースがアクティブになった後、TSO の **NETSTAT HOME** コマンドと z/OS UNIX シェルの **netstat -h** コマンドを発行して、値を表示します。TSO および z/OS UNIX シェルからホスト名の PING を発行すると、構成されている可能性がある DNS サーバーに対するアクティビティーも表示されます。

z/OS UNIX 環境で使用される検索順序

検索対象となる特定のファイルまたはテーブルは、リゾルバーの構成設定およびシステム上の所定のファイルの存在に応じて、MVS データ・セットか HFS ファイルになります。

基本リゾルバー構成ファイル

基本リゾルバー構成ファイルには、TCPIP.DATA ステートメントが入っています。このファイルは、リゾルバー・ディレクティブだけでなく、特にこのセクションで指定されている一部の構成ファイルへのアクセスを試みるときに使用されるデータ・セット接頭部 (DATASETPREFIX ステートメントの値) を判別するために、参照されます。

基本リゾルバー構成ファイルへのアクセスに使用される検索順序は、以下のとおりです。

1. GLOBALTCPIPDATA

定義されていれば、リゾルバーの GLOBALTCPIPDATA セットアップ・ステートメントの値が使用されます (222 ページの『リゾルバーについて』も参照)。検索は、追加構成ファイルについて続行されます。検索は、次のファイルが検出されると終了します。

2. 環境変数 RESOLVER_CONFIG の値

環境変数の値が使用されます。この検索は、ファイルが存在しないか他の場所で排他的に割り振られている場合、失敗します。

3. /etc/resolv.conf

4. //SYSTCPD DD カード

DD 名 SYSTCPD へ割り振られたデータ・セットが使用されます。z/OS UNIX 環境では、子プロセスは SYSTCPD DD にアクセスできません。なぜなら、SYSTCPD 割り振りは親プロセスから fork() または exec 関数呼び出しで継承されないからです。

5. userid.TCPIP.DATA

userid は、現行セキュリティ環境 (アドレス・スペース、タスク、またはスレッド) へ関連付けられているユーザー ID です。

6. jobname.TCPIP.DATA

jobname は、バッチ・ジョブの場合は JOB JCL ステートメントで指定された名前前で、開始プロシージャの場合はプロシージャ名です。

7. **SYS1.TCPPARMS(TCPDATA)**

8. **DEFAULTTCPIPDATA**

定義されていれば、リゾルバーの DEFAULTTCPIPDATA セットアップ・ステートメントの値が使用されます (222 ページの『リゾルバーについて』も参照します)。

9. **TCPIP.TCPIP.DATA**

変換テーブル

変換テーブル (EBCDIC から ASCII、および ASCII から EBCDIC) は、使用する変換データ・セットを判別するために参照されます。この構成ファイルへのアクセスに使用される検索順序は、以下のとおりです。検索順序は、最初に検出されたファイルの位置で終了します。

1. 環境変数 **X_XLATE** の値。この環境変数の値は、TSO CONVXLAT コマンドによって生成された変換テーブルの名前です。

2. **userid.STANDARD.TCPXLBIN**

userid は、現行セキュリティ環境 (アドレス・スペースまたはタスク/スレッド) へ関連付けられているユーザー ID です。

3. **jobname.STANDARD.TCPXLBIN**

jobname は、バッチ・ジョブの場合は JOB JCL ステートメントで指定された名前前で、開始プロシージャの場合はプロシージャ名です。

4. **hlq.STANDARD.TCPXLBIN**

hlq は、基本リゾルバー構成ファイル内で指定された DATASETPREFIX ステートメントがある場合は、そのステートメントの値を表します。ない場合、hlq はデフォルトの TCPIP になります。

5. テーブルが見つからない場合、リゾルバーはハードコーディングされたデフォルト・テーブルを使用します。これは、データ・セット・メンバー SEZATCPX(STANDARD) の中にリストされているテーブルと同じものです。

ローカル・ホスト・テーブル

デフォルトでは、リゾルバーは構成済みのドメイン・ネーム・サーバーがあれば、最初にそれを解決要求に使用しようとします。解決要求を満足できない場合は、ローカル・ホスト・テーブルが使用されます。リゾルバーの動作は、TCPIP.DATA ステートメントによって制御されます。

TCPIP.DATA リゾルバー・ステートメントは、ドメイン・ネーム・サーバーを使用するかどうか、およびその使用方法を定義します。LOOKUP TCPIP.DATA ステートメントは、ドメイン・ネーム・サーバーおよびローカル・ホスト・テーブルの使用方法を制御するためにも使用されます。TCPIP.DATA ステートメントの詳細については、「*Communications Server IP 構成解説書*」(SC88-8927) を参照してください。

リゾルバーは、getnetbyname API 呼び出しに対して、無条件に IPv4 固有の検索順序を使用します。sitename 情報の IPv4 固有の検索順序は、以下のとおりです。検索は、最初に検出されたファイルの位置で終了します。

1. 環境変数 **X_SITE** の値

この環境変数の値は、TSO **MAKESITE** コマンドによって作成された HOSTS.SITEINFO 情報ファイルの名前です。

2. 環境変数 **X_ADDR** の値

この環境変数の値は、TSO **MAKESITE** コマンドによって作成された HOSTS.ADDRINFO 情報ファイルの名前です。

3. **/etc/hosts**

4. **userid.HOSTS.SITEINFO**

userid は、現行セキュリティ環境 (アドレス・スペースまたはタスク/スレッド) へ関連付けられているユーザー ID です。

5. **jobname.HOSTS.SITEINFO**

jobname は、バッチ・ジョブの場合は JOB JCL ステートメントで指定された名前で、開始プロシージャの場合はプロシージャ名です。

6. **hlq.HOSTS.SITEINFO**

hlq は、基本リゾルバー構成ファイル内で指定された DATASETPREFIX ステートメントがある場合は、そのステートメントの値を表します。ない場合、hlq はデフォルトの TCPIP になります。

このセットアップ情報の Developer for System z への適用

前に述べたように、APPC を使用する場合、Developer for System z は TCP/IP が初期化時に正しいホスト名を持っているかどうか依存します。つまり、各種の TCP/IP 構成ファイルやリゾルバー構成ファイルを正しくセットアップする必要がありますということです。

以下の例では、TCP/IP およびリゾルバーのいくつかの構成タスクに焦点を当てます。これは TCP/IP またはリゾルバーの完全なセットアップについて述べたものではなく、ご使用のサイトにも適用できる可能性がある、いくつかの重要な局面だけを中心に説明したものであることに注意してください。

1. 以下の JCL では、TCP/IP が SYS1.TCPPARMS(TCPDATA) を使用してスタックのホスト名を判別することが分かります。

```
//TCPIP    PROC  PARM='CTTRACE(CTIEZB00)',PROF=TCPPROF,DATA=TCPDATA
//*
//*  TCP/IP NETWORK
//*
//TCPIP    EXEC  PGM=EZBTCPIP,REGION=0M,TIME=1440,PARM=&PARMS
//PROFILE  DD   DISP=SHR,DSN=SYS1.TCPPARMS(&PROF)
//SYSTCPD  DD   DISP=SHR,DSN=SYS1.TCPPARMS(&DATA)
//SYSPRINT DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//ALGPRINT DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//CFGPRINT DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//SYSOUT   DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//CEEDUMP  DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//SYSERROR DD   SYSOUT=*
```

2. SYS1.TCPPARMS(TCPDATA) は、システム名をホスト名にすることと、ドメイン・ネーム・サーバー (DNS) を使用しないことを示しています。すべての名前はサイト・テーブル・ルックアップによって解決されます。

```
; HOSTNAME specifies the TCP host name of this system. If not
; specified, the default HOSTNAME will be the node name specified
; in the IEFSSNxx PARMLIB member.
;
; HOSTNAME
;
; DOMAINORIGIN specifies the domain origin that will be appended
; to host names passed to the resolver. If a host name contains
; any dots, then the DOMAINORIGIN will not be appended to the
; host name.
;
DOMAINORIGIN RALEIGH.IBM.COM
;
; NSINTERADDR specifies the IP address of the name server.
; LOOPBACK (14.0.0.0) specifies your local name server. If a name
; server will not be used, then do not code an NSINTERADDR statement.
; (Comment out the NSINTERADDR line below). This will cause all names
; to be resolved via site table lookup.
;
; NSINTERADDR 14.0.0.0
;
; TRACE RESOLVER will cause a complete trace of all queries to and
; responses from the name server or site tables to be written to
; the user's console. This command is for debugging purposes only.
;
; TRACE RESOLVER
```

3. リゾルバー JCL の中で、SETUP DD ステートメントは使用されません。222 ページの『リゾルバーについて』で説明したように、これは GLOBALTCPIPDATA およびその他の変数が使用されないことを意味します。

```
//RESOLVER PROC PARMS='CTRACE(CTIRES00)'
//*
/* IP NAME RESOLVER - START WITH SUB=MSTR
/*
//RESOLVER EXEC PGM=EZBREINI,REGION=0M,TIME=1440,PARM=&PARMS
/*SETUP DD DISP=SHR,DSN=USER.PROCLIB(RESSETUP),FREE=CLOSE
```

4. RESOLVER_CONFIG 環境変数が設定されないことを想定すると、227 ページの表 40 で、リゾルバーが /etc/resolv.conf を基本構成ファイルとして使用しようとすることが分かります。

```
TCPIPJOBNAME TCPIP
DomainOrigin RALEIGH.IBM.COM
HostName CDFMVS08
```

223 ページの『z/OS UNIX 環境で使われる検索順序』で述べたように、基本構成ファイルには TCPIP.DATA ステートメントが入っています。システム名が CDFMVS08 の場合 (TCPDATA はシステム名がホスト名として使用されることを述べていました)、/etc/resolv.conf が SYS1.TCPPARMS(TCPDATA) と同期していることが分かります。DNS 定義は存在しないため、サイト・テーブル・ルックアップが使用されます。

5. 227 ページの表 40 も、デフォルトの ASCII-EBCDIC 変換テーブルを使用するために何もする必要がないことを示しています。
6. TSO **MAKESITE** コマンドが使用されない (X_SITE 変数および X_ADDR 変数を作成できる) と想定した場合、/etc/hosts が、名前のルックアップに使用されるサイト・テーブルになります。

```
# Resolver /etc/hosts file cdfmvs08
9.42.112.75    cdfmvs08                # CDFMVS08 Host
9.42.112.75    cdfmvs08.raleigh.ibm.com  # CDFMVS08 Host
127.0.0.1      localhost
```

このファイルの最低限の内容は、現行システムに関する情報です。上記の例では、z/OS システムの IP アドレスの有効な名前として、cdfmvs08 と cdfmvs08.raleigh.ibm.com の両方を定義しています。

ドメイン・ネーム・サーバー (DNS) を使用した場合は、DNS が /etc/hosts 情報を保持し、/etc/resolv.conf および SYS1.TCPPARMS(TCPDATA) には、その DNS をシステムに対して識別するステートメントが含まれることになります。

混乱を避けるために、TCP/IP 構成ファイルとリゾルバー構成ファイルを互いに同期させておく必要があります。

表 40. リゾルバーで使用可能なローカル定義

ファイル・タイプの説明	影響する API	候補ファイル
基本リゾルバー構成ファイル	すべての API	1. GLOBALTCPIPDATA 2. RESOLVER_CONFIG 環境変数 3. /etc/resolv.conf 4. SYSTCPD DD 名 5. userid.TCPIP.DATA 6. jobname.TCPIP.DATA 7. SYS1.TCPPARMS(TCPDATA) 8. DEFAULTTCPIPDATA 9. TCPIP.TCPIP.DATA
変換テーブル	すべての API	1. X_XLATE 環境変数 2. userid.STANDARD.TCPXLBIN 3. jobname.STANDARD.TCPXLBIN 4. hlq.STANDARD.TCPXLBIN 5. リゾルバー提供の変換テーブル、SEZATCPX のメンバー STANDARD

表 40. リゾルバーで使用可能なローカル定義 (続き)

ファイル・タイプの説明	影響する API	候補ファイル
ローカル・ホスト・テーブル	endhostent	IPv4 1. X_SITE 環境変数 2. X_ADDR 環境変数 3. /etc/hosts 4. userid.HOSTS.xxxxINFO 5. jobname.HOSTS.xxxxINFO 6. hlq.HOSTS.xxxxINFO
	endnetent	
	getaddrinfo	
	gethostbyaddr	
	gethostbyname	
	gethostent	
	GetHostNumber	IPv6 1. GLOBALIPNODES 2. RESOLVER_IPNODES 環境変数 3. userid.ETC.IPNODES 4. jobname.ETC.IPNODES 5. hlq.ETC.IPNODES 6. DEFAULTIPNODES 7. /etc/ipnodes
	GetHostResol	
	GetHostString	
	getnameinfo	
	getnetbyaddr	
	getnetbyname	
	getnetent	
	IsLocalHost	
	Resolve	
	sethostent	
	setnetent	

注: 227 ページの表 40は、「*Communications Server IP 構成ガイド*」(SC88-8926) の表の一部をコピーしたものです。完全な表については、その資料を参照してください。

ホスト・アドレスが正しく解決されない場合

TCP/IP リゾルバーでホスト・アドレスを正しく解決できない問題が生じた場合、原因として最も可能性が高いのは、リゾルバー構成ファイルが欠落しているか、不完全であることです。この問題を明確に示すものは、`lock.log` 内の以下のメッセージです。

```
clientip(0.0.0.0) <> callerip(<host IP address>)
```

これを検査するには、「*ホスト構成ガイド*」(SC88-5663)の"『インストール検査』"の説明に従って、`fekfivpt TCP/IP IVP` を実行します。出力のリゾルバー構成セクションは、以下のサンプルのようになります。

```
Resolver Trace Initialization Complete -> 2008/07/02 13:11:54.745964
```

```
res_init Resolver values:
Global Tcp/Ip Dataset = None
Default Tcp/Ip Dataset = None
Local Tcp/Ip Dataset = /etc/resolv.conf
Translation Table = Default
UserId/JobName = USERID
Caller API = LE C Sockets
Caller Mode = EBCDIC
```

「Local Tcp/Ip Dataset」が示すファイル (データ・セット) 内の定義が正しいことを確認してください。

このフィールドは、IP リゾルバー・ファイルに (z/OS UNIX 検索順序を使用して) デフォルト名を使用していない場合にはブランクになります。その場合は、次のステートメントを `rsed.envvars` に追加します。ここで、`<resolver file>` または `<resolver data>` は IP リゾルバー・ファイルの名前を表しています。

```
RESOLVER_CONFIG=<resolver file>
```

or

```
RESOLVER_CONFIG='<resolver data set>'
```


参考文献

参考資料

本書では、以下の資料を参照しています。

表 41. 参考資料

資料名	資料番号	参照	参照 Web サイト
IBM Rational Developer for System z Program Directory	GI88-4172	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Program Directory for IBM Rational Developer for System z Host Utilities	GI13-2864	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System z前提条件	SC88-4704	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System z ホスト構成クイック・スタート・ガイド	GI88-4171	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System z ホスト構成ガイド	SC88-5663	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System zホスト構成リファレンス	SA88-4226	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System z ホスト構成ユーティリティー・ガイド	SA88-4197	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System z メッセージとコード・ガイド	SA88-4565	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System z共通ホスト構成と保守に関する問題への回答	SC14-7373	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System z Common Access Repository Manager Developer's Guide	SC23-7660	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Rational Developer for System z前提条件	SC88-4704	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z ホスト構成クイック・スタート・ガイド	GI88-4171	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
SCLM Developer Toolkit 管理者ガイド	SC88-5664	Developer for System z	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Using APPC to provide TSO command services	SC14-7291	ホワイト・ペーパー	http://www-01.ibm.com/support/docview.wss?uid=swg27038517

表 41. 参考資料 (続き)

資料名	資料番号	参照	参照 Web サイト
Using ISPF Client Gateway to provide CARMA services	SC14-7292	ホワイト・ペーパー	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Communications Server IP 構成ガイド	SC88-8926	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server IP 構成解説書	SC88-8927	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server IP Diagnosis Guide	GC31-8782	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server IP システム管理者のコマンド	SC88-9073	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server SNA ネットワーク・インプリメンテーション・ガイド	SC88-8928	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server SNA オペレーション	SC88-8930	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Cryptographic Services System SSL (Secure Sockets Layer) プログラミング	SD88-6252	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
DFSMS Macro Instructions for Data Sets	SC26-7408	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
DFSMS データ・セットの使用法	SC88-9114	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
言語環境プログラム カスタマイズ	SA88-8552	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
言語環境プログラム デバッグ・ガイド	GA88-8548	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 診断: ツールと保守援助プログラム	GA88-8561	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 初期設定およびチューニング ガイド	SA88-8563	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 初期設定およびチューニング解説書	SA88-8564	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS JCL 解説書	SA88-8569	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 計画 APPC/MVS 管理	SA88-8571	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 計画: ワークロード管理	SA88-8574	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS システム・コマンド	SA88-8593	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Security Server RACF コマンド言語解説書	SA88-8617	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Security Server RACF セキュリティー管理者のガイド	SA88-8613	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
TSO/E カスタマイズ	SA88-8629	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
TSO/E REXX 解説書	SA88-8635	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/

表 41. 参考資料 (続き)

資料名	資料番号	参照	参照 Web サイト
UNIX System Services コマンド解説書	SA88-8641	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
UNIX System Services 計画	GA88-8639	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
UNIX システム・サービス ユーザーズ・ガイド	SA88-8640	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
REXX および z/OS UNIX システム・サービスの使い方	SA88-8644	z/OS 1.13	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Java™ Diagnostic Guide	SC34-6650	Java 6.0	http://www.ibm.com/developerworks/java/jdk/diagnosis/
Java SDK and Runtime Environment User Guide	/	Java 6.0	http://www-03.ibm.com/servers/eserver/zseries/software/java/
Resource Definition Guide	SC34-6430	CICSTS 3.1	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
Resource Definition Guide	SC34-6815	CICSTS 3.2	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
Resource Definition Guide	SC34-7000	CICSTS 4.1	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
Resource Definition Guide	SC34-7181	CICSTS 4.2	https://publib.boulder.ibm.com/infocenter/cicsts/v4r2/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
RACF Security Guide	SC34-6454	CICSTS 3.1	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
RACF Security Guide	SC34-6835	CICSTS 3.2	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
RACF Security Guide	SC34-7003	CICSTS 4.1	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
RACF Security Guide	SC34-7179	CICSTS 4.2	https://publib.boulder.ibm.com/infocenter/cicsts/v4r2/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
言語解説書	SC88-9117	Enterprise COBOL for z/OS	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html

本書では、以下の Web サイトを参照しています。

表 42. 参照される Web サイト

説明	参照 Web サイト
Developer for System z インフォメーション・センター	http://pic.dhe.ibm.com/infocenter/ratdevz/v9r0/index.jsp
Developer for System z ライブラリー	http://www-01.ibm.com/support/docview.wss?uid=swg27038517
Developer for System z ホーム・ページ	http://www-03.ibm.com/software/products/us/en/developerforsystemz/
Developer for System z 推奨サービス	http://www-01.ibm.com/support/docview.wss?rs=2294&context=SS2QJ2&uid=swg27006335

表 42. 参照される Web サイト (続き)

説明	参照 Web サイト
Developer for System z 機能拡張要求	https://www.ibm.com/developerworks/support/rational/rfe/
z/OS インターネット・ライブラリー	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
CICSTS インフォメーション・センター	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp
IBM Tivoli Directory Server	http://www-01.ibm.com/software/tivoli/products/directory-server/
Problem Determination Tools Plug-ins	http://www-01.ibm.com/software/awdtools/deployment/pdtpplugins/
Apache Ant のダウンロード	http://ant.apache.org/
Java keytool の資料	http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html
CA サポート・ホーム・ページ	https://support.ca.com/

情報資料

以下の資料は、必要なホスト・システム・コンポーネントのセットアップの問題を理解するのに役立ちます。

表 43. 情報資料

資料名	資料番号	参照	参照 Web サイト
ABCs of z/OS System Programming Volume 9 (z/OS UNIX)	SG24-6989	Redbook	http://www.redbooks.ibm.com/
System Programmer's Guide to: Workload Manager	SG24-6472	Redbook	http://www.redbooks.ibm.com/
TCPIP Implementation Volume 1: Base Functions, Connectivity, and Routing	SG24-7532	Redbook	http://www.redbooks.ibm.com/
TCPIP Implementation Volume 3: High Availability, Scalability, and Performance	SG24-7534	Redbook	http://www.redbooks.ibm.com/
TCP/IP Implementation Volume 4: Security and Policy-Based Networking	SG24-7535	Redbook	http://www.redbooks.ibm.com/
Tivoli Directory Server for z/OS	SG24-7849	Redbook	http://www.redbooks.ibm.com/

用語集

アクション ID (Action ID)

アクションを表す 0 から 999 までの数値 ID。

アプリケーション・サーバー (Application Server)

1. ブラウザー・ベースのコンピューターと組織のバックエンド・ビジネス・アプリケーションまたはデータベースとの間で、すべてのアプリケーション操作を処理するプログラム。Java EE 規格に準拠した、Java ベースの特殊クラスの appserver が存在する。Java EE コードは、これらのアプリケーション・サーバー間での移植が容易である。動的 Web コンテンツ用の JSP とサーブレット、およびトランザクションとデータベース・アクセス用の EJB をサポートできる。
2. リモート・アプリケーションからの要求のターゲット。DB2 環境において、アプリケーション・サーバー機能は分散データ機能によって提供され、リモート・アプリケーションの DB2 データにアクセスするために使用される。
3. アプリケーション・プログラムの実行環境を提供する分散ネットワーク内のサーバー・プログラム。
4. アプリケーション・リクエスターからの要求のターゲット。アプリケーション・サーバー・サイトのデータベース管理システム (DBMS) は、要求されたデータを提供する。
5. Content Manager のアセットおよび照会を要求するクライアントとの通信を処理するソフトウェア。

双方向 (Bidirectional) (bi-di)

数字 (左から右に書かれる) を除き、一般に右から左に書かれるアラビア語やヘブライ語などのスクリプトに関する用語。この

定義は、Localization Industry Standards Association (LISA) の用語集からのものである。

双方向属性 (Bidirectional Attribute)

テキスト・タイプ、テキスト方向、数値スワッピング、および対称スワッピング。

ビルド要求 (Build Request)

ビルド・トランザクションの実行を求めるクライアントからの要求。

ビルド・トランザクション (Build Transaction)

クライアントからビルド要求を受信した後に、MVS 上で開始されるジョブ。

コンパイル (Compile)

1. 統合化言語環境 (Integrated Language Environment) (ILE) 言語において、ソース・ステートメントを、プログラムまたはサービス・プログラムにバインドできるモジュールに変換すること。
2. 高水準言語で表現されたプログラムの全部または一部を、中間言語、アセンブリ言語、またはマシン言語で表現されたコンピューター・プログラムに変換すること。

コンテナ (Container)

1. CoOperative Development Environment/400 においては、ソース・ファイルを格納および編成するシステム・オブジェクト。コンテナの例としては、i5/OS™ ライブラリーや MVS 区分データ・セットがある。
2. Java EE において、コンポーネントにライフサイクル管理、セキュリティ、デプロイメント、およびランタイム・サービスを提供するエンティティ。 (Sun) 各タイプのコンテナ (EJB、Web、JSP、サーブレット、アプレット、およびアプリケーション・クライアント) はコンポーネント固有のサービスも提供する。

3. バックアップ・リカバリーおよびメディア・サービスにおいて、ボックス、ケース、またはラックなどのメディアの保管と移動に使用される物理オブジェクト。
4. 仮想テープ・サーバー (VTS) において、エクスポートされた 1 つ以上の論理ボリューム (LVOL) を保管できる貯蔵所。1 つ以上の LVOL を含み、VTS ライブラリーの外部に存在するボリューム・スタックは、そのボリュームのコンテナと見なされる。
5. データの物理的な記憶場所。例えば、ファイル、ディレクトリー、デバイスなど。
6. ポートレットまたはページ上にあるその他のコンテナのレイアウトを調整するために使用される列または行。
7. オブジェクトを保持するユーザー・インターフェースの要素。フォルダー・マネージャーにおいて、他のフォルダーまたは文書を格納できるオブジェクト。

データベース (Database)

1 つ以上のアプリケーションに対してサービスを行うために、まとめて保管される、相互に関連するか独立したデータ項目のコレクション。

データ定義ビュー (Data Definition View)

データベースとそのオブジェクトのローカル表現が入っており、それらのオブジェクトを操作してリモート・データベースにエクスポートする機能を提供する。

データ・セット (Data Set)

データの保管と取り出しの主要単位。いくつかの規定された配置の 1 つに置かれたデータのコレクションで構成され、システムがアクセスする制御情報によって記述される。

デバッグ (Debug)

プログラム内のエラーを検出、診断、および除去すること。

デバッグ・セッション (Debugging Session)

開発者がデバッガーを開始した時点から、

デバッガーを終了する時点までに発生するデバッグ・アクティビティー。

エラー・バッファー (Error Buffer)

エラー出力情報を一時的に保持するために使用されるストレージの部分。

ゲートウェイ (Gateway)

1. Web サービス呼び出しのとき、インターネットとイントラネット環境のブリッジとなるミドルウェア・コンポーネント。
2. エンドポイントとそれ以外の Tivoli 環境の間でサービスを提供するソフトウェア。
3. Voice over Internet Protocol 環境と回路交換環境の間のブリッジとなる VoIP のコンポーネント。
4. 異なるネットワーク・アーキテクチャーを備えたネットワーク、またはシステムを接続するために使用されるデバイスまたはプログラム。各システムが異なる特性 (例えば、異なる通信プロトコル、異なるネットワーク・アーキテクチャー、異なるセキュリティ・ポリシーなど) を備えている場合があり、ゲートウェイは、そのような場合に変換の役割と接続の役割を果たす。

対話式システム生産性向上機能 (Interactive System Productivity Facility) (ISPF)

フルスクリーン・エディターおよびダイアログ・マネージャーとして機能する IBM ライセンス・プログラム。アプリケーション・プログラムの作成に使用され、標準的な画面パネルとアプリケーション・プログラマーと端末ユーザーとの間に対話式ダイアログを生成する手段となる。ISPF は、DM、PDF、SCLM、および C/S という 4 つの主要なコンポーネントからなる。DM

コンポーネントとはダイアログ・マネージャーのことで、ダイアログとエンド・ユーザーにサービスを提供する。PDF コンポーネントとはプログラム開発機能のことで、ダイアログまたはアプリケーション開発者を支援するサービスを提供する。

SCLM コンポーネントとは Software Configuration Library Manager のことで、アプリケーション開発者にアプリケーション開発ライブラリーを管理するためのサービスを提供する。C/S コンポーネントとは、クライアント/サーバーのことで、これによってプログラマブル・ワークステーション上で ISPF を実行し、ワークステーションのオペレーティング・システムの表示機能を使用してパネルを表示し、ワークステーションのツールおよびデータをホストのツールおよびデータと統合することができる。

インタープリター (Interpreter)

高水準プログラミング言語の 1 つの命令を翻訳および実行してから、次の命令を翻訳および実行するプログラム。

アイソモアフィック (Isomorphic)

ルートから始まる XML インスタンス文書の構成された各エレメント (つまり、他のエレメントを含んでいるエレメント) は、唯一の対応する COBOL グループ項目を持ち、そのグループ項目のネストの深さは、同等の XML 項目のネストの深さと同一である。トップから始まる XML インスタンス文書の構成されていない各エレメント (つまり、他のエレメントを含んでいないエレメント) は、唯一の対応する COBOL 基本項目を持っており、その基本項目のネストの深さは、同等の XML 項目のネスト・レベルと同じであり、実行時のメモリー・アドレスは、一意に識別できる。

リンケージ・セクション (Linkage Section)

アクティブにされる単位 (呼び出されたプログラムまたはメソッド) のデータ部に含まれており、アクティブにする単位 (プログラムまたはメソッド) から使用可能なデータ項目を記述しているセクション。これらのデータ項目は、アクティブにされる単位とアクティブにする単位の両方から参照できる。

ロード・ライブラリー (Load Library)

ロード・モジュールを含んでいるライブラリー。

ロック・アクション (Lock Action)

メンバーをロックする。

ナビゲーター・ビュー (Navigator View)

ワークベンチ内のリソースの階層図を提供する。

非アイソモアフィック (Non-Isomorphic)

形態が同一でない (非アイソモアフィック) XML 文書と COBOL グループに属する、COBOL 項目と XML エレメントとの単純なマッピング。非アイソモアフィック・マッピングは、アイソモアフィック構造の非アイソモアフィック・エレメント間でも作成できる。

「出力コンソール」ビュー (Output Console View)

プロセスの出力を表示し、プロセスへのキーボード入力を提供する。

出力ビュー (Output View)

処理対象のオブジェクトに関連したメッセージ、パラメーター、および結果を表示する。

パースペクティブ (Perspective)

ワークベンチ内のリソースのさまざまな側面を表示するビューのグループ。ワークベンチ・ユーザーは、手元のタスクに応じてパースペクティブを切り替え、パースペクティブ内のビューおよびエディターのレイアウトをカスタマイズできる。

RAM Repository Access Manager の略。

リモート・ファイル・システム (Remote File System)

別のサーバーまたはオペレーティング・システム上にあるファイル・システム。

リモート・システム (Remote System)

ネットワーク内にあり、使用しているシステムの通信相手にできる別のシステム。

「リモート・システム」パースペクティブ (Remote Systems Perspective)

ISPF に似た規約を使用して、リモート・システムを管理するインターフェースを提供する。

リポジトリ (Repository)

1. データのストレージ域。すべてのリポジトリは名前と、関連するビジネス項目タイプを備えている。デフォルトでは、名前はビジネス項目の名前と同じものになる。例えば、送り状のリポジトリは「Invoices」と呼ばれる。ローカル (プロセスに固有のもの) とグローバル (再利用可能なもの) の 2 つのタイプの情報リポジトリがある。
2. BTS プロセスの状態を保管している VSAM データ・セット。プロセスが BTS の制御下で実行されていない場合、そのプロセスの状態 (およびそのプロセスを構成するアクティビティの状態) は、リポジトリ・データ・セットに書き込まれることによって保存される。特定のプロセス・タイプに属するすべてのプロセス (およびそれらのアクティビティ・インスタンス) の状態は、同じリポジトリ・データ・セットに保管される。複数のプロセス・タイプのレコードを同じリポジトリに書き込むことができる。
3. ソース・コードおよびその他のアプリケーション・リソース用の永続的なストレージ域。チーム・プログラミング環境においては、共用リポジトリに

よってアプリケーション・リソースへのマルチユーザー・アクセスが可能になる。

4. クラスターのメンバーであるキュー・マネージャーに関する情報のコレクション。この情報には、キュー・マネージャーの名前、ロケーション、チャンネル、ホスト対象となるキューなどが含まれる。

リポジトリ・インスタンス (Repository Instance)

SCM 内に存在するプロジェクトまたはコンポーネント。

リポジトリ・ビュー (Repositories View)

ワークベンチに追加された CVS リポジトリ・ロケーションを表示する。

応答ファイル (Response File)

1. プログラムからの質問に対する定義済みの回答セットが入っているファイル。それらの値を一度に 1 つずつ入力する代わりに使用される。
2. インストールを自動化するセットアップおよび構成データを使用してカスタマイズできる ASCII ファイル。セットアップおよび構成データは本来、対話式インストールのときに入力すべきものだが、応答ファイルを使用すると、インストールを介入なしに進行させることができる。

サーバー・ビュー (Servers View)

使用しているすべてのサーバーとそれに関連した構成をリストとして表示する。

シェル (Shell)

ユーザーとオペレーティング・システム間のソフトウェア・インターフェース。コマンドおよびユーザー対話を解釈し、ユーザーとオペレーティング・システム間の通信を行う。さまざまなレベルのユーザー対話を処理するために、1 つのコンピュータに、複数のレイヤーのシェルが存在する場合がある。

シェル名 (Shell Name)

シェル・インターフェースの名前。

シェル・スクリプト (Shell Script)

シェルが解釈できる、コマンドが入ったファイル。ユーザーは、シェル・コマンド・プロンプトでスクリプト・ファイルの名前を入力して、シェルにスクリプト・コマンドを実行させる。

SIDEDECK

DLL プログラムの機能を公開するライブラリー。ソース・コードがコンパイルされた後、入り口名とモジュール名は、このライブラリーに保管される。

サイレント・インストール (Silent Installation)

コンソールにメッセージを送らず、その代わりにメッセージとエラーをログ・ファイルに保管するインストール。また、サイレント・インストールはデータ入力に応答ファイルを使用できる。

サイレント・アンインストール (Silent Uninstallation)

アンインストール・コマンドが起動された後、コンソールにメッセージを送らず、その代わりにメッセージとエラーをログ・ファイルに保管するアンインストール・プロセス。

タスク・リスト (Task List)

単一の制御フローによって実行できる手順のリスト。

URL Uniform Resource Locator の略。

IBM Rational Developer for System z 資料に関する特記事項

© Copyright IBM Corporation 2009, 2013.

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒103-8510
東京都中央区日本橋箱崎町19番21号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

Intellectual Property Dept. for Rational Software
IBM Corporation
5 Technology Park Drive
Westford, MA 01886
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性があります。その測定値が、一般に利用可能なシステムのもと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。サンプル・プログラムは、現存するままの状態を提供され、いかなる保証条件も適用されません。IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年).このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 © Copyright IBM Corp. 2009, 2013.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

商標の帰属表示

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、www.ibm.com/legal/copytrade.shtml をご覧ください。

Adobe、Adobe ロゴ、PostScript、PostScript ロゴは、Adobe Systems Incorporated の米国およびその他の国における登録商標または商標です。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Windows は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。

著作権使用許諾

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプ

リケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほめめかしたり、保証することはできません。サンプル・プログラムは、現存するままの状態を提供され、いかなる保証条件も適用されません。IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

商標の帰属表示

IBM、IBM ロゴおよび `ibm.com` は、世界の多くの国で登録された International Business Machines Corp. の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、www.ibm.com/legal/copytrade.shtml をご覧ください。

CA Endeavor は CA Technologies の登録商標です。

Intel および Pentium は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Microsoft、Windows および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

【ア行】

アクション、ジョブに対する - 実行の制限 31
アクセス、スプール・ファイルへの条件付き 32
アクセスの改善、システム・ライブラリーへの 124
アクセス方式、TSO 171
アクセス方式の使用、TSO/ISPF クライアント・ゲートウェイ 172
アクティベーション 136
アドレス・スペースの数 89
アドレス・スペース・サイズ 201
アドレス・スペース・サイズの限度 100
アプリケーション開発 124
アプリケーション保護の定義、RSE の 55
暗号化、SSL または TLS 206
暗号化、SSL を使用した通信 24
暗号化、TLS を使用した通信 24
暗号化通信
統合デバッガー 34
暗号化通信、SSL 43, 155
暗号化通信、SSL/TLS 33
依存関係、ホスト名 221
一時的なリソース使用量 98
インストール・ロギング、CICS リソース 153
エラー・フィードバック・トレース 195

【カ行】

解決されないホスト・アドレス、TCP/IP リゾルバー
lock.log 228
開始タスクの定義、Developer for System z
JMON 開始タスク 51
RSED 開始タスク 51
改善、システム・ライブラリーへのアクセスの 124
開発、アプリケーション 124
外部通信 66
外部通信の限定、指定したポート 24

概要、クライアントへのプッシュの考慮事項 131
鍵ストアの作成、keytool による 217
鍵データベースの作成、gskkyman による 215
鍵リングの作成、RACF による 207
確認応答、遅延 68
カスタマイズ - ISPF.conf 172
カスタマイズ、Application Deployment Manager 151
カスタマイズ、TSO 環境の 171
可変ユーザー ID、実行 166
監査処理
modify switch 27
監査制御
audit.* options 27
daemon.log 27
enable.audit.log 27
_RSE_HOST_CODEPAGE 27
監査データ
ログに記録されるアクション 28
監査ロギング、RSE デーモンで管理 27
管理、ワークロード 125
管理ユーティリティ、マイグレーションに関する注 159
管理ユーティリティのメッセージ 160
基本リゾルバー構成ファイル 223
キャッシュ管理ユーティリティ、Java 仮想マシン (JVM) 128
キャッシュ・サイズ制限、Java 仮想マシン (JVM) 127
キャッシュ・セキュリティ、Java 仮想マシン (JVM) 127
共存、共存を可能にするための
rsed.envvars の更新 210
共通アクセス・リポジトリ・マネージャー・ロギング 188
許可ビット、z/OS UNIX 196
拒否、接続の 203
クイック・スタート、Java オプション (-Xquickstart) 126
クライアント関数、変更 39
クライアント構成の制御 135
クライアント認証、X.509 証明書を使用した 35
クライアント認証サポートの追加、X.509 214
クライアントへのプッシュ 41
クライアントへのプッシュの考慮事項 131

クライアントへのプッシュ・バックエンド、LDAP への追加 143
クライアントへのプッシュ・メタデータ 133
クライアント・ゲートウェイ・アクセス方式の使用、TSO/ISPF 172
クライアント・バージョンの制御 135
クラス共用、Java 仮想マシン (JVM) 間 126
クラス共用、Java 仮想マシン (JVM) での有効化 127
クラス共用の有効化、Java 仮想マシン (JVM) 127
グループ選択、LDAP ベース 140
グループ選択、SAF ベース 145
グループの連結 137
グループ・メタデータのロケーション 138
クローン作成、既存の RSE セットアップ 209
言語環境プログラム・ランタイム・ライブラリー 124
検査、セキュリティ設定 63
検索順序、構成情報の 222
検索順序、z/OS UNIX 環境 223
限度、システム 203
コード・カバレッジのロギング 190
コード・レビューのロギング 190
向上、セキュリティ検査のパフォーマンスの 125
更新特権、非システム管理者 17
構成情報の検索順序 222
構成ファイル、基本リゾルバー 223
構成ファイル、同一のソフトウェア・レベルの、異なる 178
構成ファイル、Developer for System z 44
構成問題のトラブルシューティング 183
考慮事項、セキュリティに関する 21
考慮事項、パフォーマンスに関する 123
固定 Java ヒープ・サイズ 126
異なる構成ファイル、同一のソフトウェア・レベル 178
コマンド・セキュリティの定義、JES 56
コンソール・メッセージ、ユーザー出口 166
コンポーネントの概要、Developer for System z
グラフィカル表現 4

[サ行]

サード・パーティーおよび X.509 証明書 22

サーバーの選択、LDAP 141

サーバーのロケーション、LDAP 142

サイズ、アドレス・スペース 201

サイズの限度、アドレス・スペース 100

サイズの限度、Java ヒープ 99

サイズ見積もり、ガイドライン 100

サブシステム・タイプ

- ASCH 78
- CICS 78
- JES 78
- OMVS 78
- STC 78

サポートの定義、RSE の PassTicket 54

さまざまなリソース定義 112

- EXEC カード、サーバー JCL 112
- FEJCNFG 112
- SYS1.PARMLIB(ASCHPMxx) 113
- SYS1.PARMLIB(IEASYXxx) 113
- SYS1.PARMLIB(IVTPRMxx) 113

参考資料 231

サンプルのストレージ、使用量分析 101

サンプル・セットアップ 118

- 限度の定義 119
- 最小限度の特定 119
- スレッド・プールの数 118

サンプル・セットアップ、LDAP グループ選択 142

サンプル・セットアップ、SAF ベースのグループ選択 146

システム限度 203

システム出口によって強制される制限 202

システム・ライブラリーへのアクセスの改善 124

シスプレックス、同一セットアップ 177

実行、複数のインスタンスの 177

実行の制限、ジョブに対するアクション 31

指定したポートとの外部通信、限定 24

始動 JCL の要件 201

自動同期 179

主接続領域と非主接続領域 152

主要なリソース定義 108

- rsed.envvars 108
- SYS1.PARMLIB(BPXPRMxx) 109

使用、既存の ISPF プロファイルの 172

使用、割り振り exec の 173

使用、PassTicket 25

照会、証明書失効リスト (CRL)

- CRL 環境変数 36
- rsed.envvars 36

条件付きアクション、ジョブに対する 29

条件付きアクセス、スプール・ファイルへの 32

使用の回避、STEPLIB の 123

証明書、X.509 22

証明書、X.509 を使用したクライアント認証 35

証明書失効リスト (CRL) の照会

- CRL 環境変数 36
- rsed.envvars 36

使用量分析、サンプルのストレージ 101

ジョブに対する条件付きアクション 29

署名付き証明書、自己署名または認証局による署名 208

資料、参考 231

スティッキー・ビット、z/OS UNIX で MVS ロード・モジュールを使用可能にする 199

ストレージの使用量 99

スプール・ファイルへの条件付きアクセス 32

スペース使用量、メタデータ 134

スペースの使用量、z/OS UNIX ファイル・システム 105

スレッドの数 95

スレッド・プール、ロギング 186

制御ライブラリーの定義、RSE の MVS 53

セキュアな z/OS UNIX サーバーとしての RSE の定義 52

セキュリティ、接続 23

セキュリティ、デバッグ 42

セキュリティ、トランザクション 154

セキュリティ、パイプライン 153

セキュリティ、リソース 155

セキュリティ、Application Deployment Manager (ADM) 153

セキュリティ、CICSTS 43

セキュリティ、JES 29

セキュリティ、SCLM 44

セキュリティ検査のパフォーマンスの向上 125

セキュリティ設定、検査 63

セキュリティ定義 47, 147

セキュリティ定義、チェックリスト 48

セキュリティに関する考慮事項 21

セキュリティの設定およびクラスのアクティブ化 49

セキュリティの定義、JES コマンド 56

セキュリティ・コマンド、便利

- ADDGROUP 17
- ALTUSER 17
- CONNECT 17

セキュリティ・ソフトウェアによる認証 37

セキュリティ・プロファイル内に保管される制限 201

セグメントの定義、OMVS 50

接続セキュリティ 23

接続の拒否 203

接続のフロー 8

- グラフィカル表現 8

接続領域、主と非主 152

設定およびクラスのアクティブ化、セキュリティの 49

セットアップ、シスプレックス全体で同一 177

セットアップ手順 139

説明、Developer for System z 3

ソフトウェア・レベル、異なる構成ファイルで同一 178

[タ行]

タスク所有者 6

ダンプ、Java 191

ダンプ、MVS 190

ダンプ・ファイル 190

ダンプ・ロケーション、z/OS UNIX 192

遅延 ACK 68

チューニングに関する考慮事項 87

通信、外部 66

通信、内部 66

通信、SSL 暗号化 155

通信、SSL/TLS 暗号化 33

通信暗号化、SSL を使用した 24

通信暗号化、TLS を使用した 24

データ・セット・プロファイルの定義 58

テーブル、変換 224

テーブル、ローカル・ホスト 224

定義、セキュリティ 47

定義、リゾルバーで使用可能な 227

定義、RSE の MVS プログラム制御ライブラリー 53

定義、RSE の PassTicket サポート 54

定義、RSE の z/OS UNIX プログラム制御ファイル 62

ディスク・スペース、Java 仮想マシン (JVM) 128

ディレクトリー構造、z/OS UNIX

- グラフィカル表現 15

出口点、使用可能な 168

テスト、SSL ホスト構成接続の 211

テスト・ロギング、fekfivpc IVP 189

テスト・ロギング、fekfivpi IVP 189

デバッガー、統合 10

デバッグ、CICS トランザクション 163

デバッグ・セキュリティ 42

デバッグ・マネージャーでの認証 23

同一セットアップ、シスプレックス全体 177

同一のソフトウェア・レベル、異なる構成
ファイル 178
同期、自動 179
統合デバッガー 10
暗号化通信 34
特定のポート、外部通信の限定 24
トラブルシューティング、構成問題の
183
トランザクション・セキュリティ 154
トランザクション・ダンプ・パターン変数
191
トレース 193
トレース、エラー・フィードバック 195
トレース、CARMA 194
トレース、JES ジョブ・モニターの 193
トレース、RSE 193

[ナ行]

内部通信 66
認証、セキュリティ・ソフトウェアによ
る 37
認証、デバッグ・マネージャー 23
認証、JES ジョブ・モニター 23
認証、RSE デーモンによる 38
認証、SSL および X.509 のセットアップ
205
認証局の妥当性検査
gskkyman 35
SAF 鍵リング 35
TRUST、HIGHTRUST 35
認証方式 22
ネットワークのモニター 117

[ハ行]

パイプライン・セキュリティ 153
場所、秘密鍵と証明書を保管する 206
パスワードおよびユーザー ID 22
パフォーマンスに関する考慮事項 123
パフォーマンスの向上、セキュリティ検
査の 125
ヒープ・サイズの限度、Java 99
非システム管理者の更新特権 17
秘密鍵と証明書を保管する場所の決定
206
ファイル・システム、zFS 123
ファイル・システム属性、SETUID 196
ファイル・システム・スペースの使用量、
z/OS UNIX 105
フィードバック・トレース、エラー 195
複数の Developer for System z セットア
ップでの複数の ISPF.conf ファイルの使
用 174
複数の ISPF.conf ファイル 174

複数のインスタンスの実行 177
複数の開発者グループ 136
複数の割り振り exec の使用、
TSO/ISPF 174
プログラム制御許可 196
プロジェクト、ホストベース 148
プロセスの数 92
プロファイルの定義、データ・セット 58
分類規則、WLM 78
変換テーブル 224
変更の拒否、猶予期間 148
ポート、予約済み TCP/IP 199
ポート、CARMA と TCP/IP 67
ポート、TCP/IP 65
ポート選択、制限 72
ポートの予約、TCP/IP 67
方式、認証 22
ホスト名、Developer for System z への適
用 225
ホスト名依存関係 221
ホスト・テーブル、ローカル 224
ホスト・ベースのプロジェクト 148

[マ行]

マイグレーションに関する注、管理ユーテ
ィリティ 159
メタデータ、クライアントへのプッシュ
133
メタデータのスペース使用量 134
メタデータのセキュリティ 133
メタデータのロケーション 133
メッセージ、管理ユーティリティ 160
メモリー不足エラー 203
目標、WLM での設定 79
目標の設定、WLM 79
モニター、ネットワーク 117
モニター、RSE 114
モニター、z/OS UNIX 115
モニター、z/OS UNIX ファイル・システ
ム 118

[ヤ行]

ユーザー ID およびパスワード 22
ユーザー ID およびワнтаイム・パスワ
ード 22
ユーザー出口、コンソール・メッセージ
166
ユーザー出口点、使用可能な 168
ユーザー出口に関する考慮事項 xv, 165
ユーザー出口の活動化 165
ユーザー出口の特性 165
ユーザー出口ルーチンの作成 165
ユーザー・ロギング、RSE 187

猶予期間、変更の拒否 148
要件、始動 JCL の 201
予約、TCPIP ポート 67
予約済み TCP/IP ポート 199

[ラ行]

ライブラリー、言語環境プログラム・ラン
タイム 124
ライブラリーの定義、RSE の MVS 53
ライブラリーへのアクセスの改善、システ
ム 124
ランタイム・ライブラリー、言語環境プロ
グラム 124
リソース使用量、一時的な 98
リソース使用量、概要 88
リソース使用量、チューニング 87
リソース定義、さまざまな 112
リソース・インストール・ロギング、
CICS 153
リソース・セキュリティ 155
リゾルバーで使用可能なローカル定義
227
リゾルバーについて 222
リポジトリ・セキュリティ、
CRD 153
ローカル・ホスト・テーブル 224
ロギング、コード・カバレッジ 190
ロギング、コード・レビュー 190
ロギング、スレッド・プール 186
ロギング、CARMA 188
ロギング、fekfivpi IVP テスト 189
ロギング、JES ジョブ・モニター 186
ロギング、RSE デーモン 186
ロギング、RSE ユーザー 187
ロギング、SCLM Developer Toolkit 188
ログとセットアップの分析に使用、
FEKLOGS 184
ログ・ファイル
audit.log 184
fa.log 184
fekfivpi.log 184
fekfivps.log 184
ffsget.log 184
ffsput.log 184
ffs.log 184
lock.log 184
rmt_class_loader.cache.jar 184
rsecomm.log 184
rsedaemon.log 184
rseserver.log 184
serverlogs.count 184
stderr.log 184
stdout.log 184
.dstoreMemLogging 184
.dstoreTrace 184

ロックの解放

RSE、modify cancel コマンド 14

ロック・デーモン 13

ロック・デーモン (LOCKD) 4

ロック・デーモンのフロー

グラフィカル表現 13

[ワ行]

ワークスペースのバインディング 137

ワークロード管理 125

ワークロード分類、WLM 77

ワークロード・マネージャー 77

割り振り exec の使用 173

ワнтаイム・パスワードおよびユーザー

ID 22

[数字]

1 次システム 132

A

ACEE、管理 44

ACK、遅延 68

ADNJSPAU、管理ユーティリティ 155

APF 許可

FEK.SFEKAUTH 58

APF、許可 198

Application Deployment Manager

(ADM) 4

Application Deployment Manager セキュリ

ティー 153

Application Deployment Manager、カスタ

マイズ 151

Application Deployment Manager、CICS リ

ソース定義エディター 151

Application Deployment Manager、CICS リ

ソース定義サーバー 151

AQEZPCM 23

ASCHPMxx

MAX 113

ASSIZEMAX 52

audit.action、ユーザー出口 168

audit.log 185

B

BPXPRMxx 120

INADDRANYCOUNT 112

MAXASSIZE 52, 110, 201

MAXFILEPROC 110

MAXMMAPAREA 110

MAXPROCSYS 109, 203

MAXPROCUSER 109, 203

BPXPRMxx (続き)

MAXSOCKETS 112

MAXTHREADS 109

MAXTHREADTASKS 109

MAXUIDS 110, 203

C

CARMA と TCP/IP ポート 67

CARMA トレース 194

CARMA ロギング

rsecomm.log 188

CEE.SCEELPA

SYS1.PARMLIB(LPALSTxx) 124

CICS 管理者向け管理ユーティリティ

提供されている機能 155

CICS トランザクション 43

CICS トランザクションのデバッグ 163

CICS リソース定義 (CRD) エディター、

Application Deployment Manager 151

CICS リソース定義 (CRD) サーバー、

Application Deployment Manager 151

CICS リソース定義、開発者 151

CICS リソース定義、管理者 151

CICS リソース・インストール・ロギング

153

CICSplex SM ビジネス・アプリケーション・サービス (BAS) 152

CICSTS セキュリティ 43

CICSTS に関する考慮事項 151

CLASSPATH 179

COBOL

リモート検査 195

CRD リポジトリ 43

CRD リポジトリ・セキュリティ

153

D

Developer for System z 開始タスク、定義

51

Developer for System z について 3

Developer for System z、コンポーネント

の概要

グラフィカル表現 4

Distributed Dynamic VIPA

EZBEPOR 70

PORT 70

PORTRANGE 70

SERVERWLM 70

SYSPLEXPORTS 70

VIPADISTRIBUTE 70

E

Emulator、Host Connect 204

F

fa.log 184

FEJCNFG 66, 120, 181

CONSOLE_NAME 31

MAX_THREADS 112

FEJCNFG、JES ジョブ・モニター 44

FEKAPPL 23

fekfivpc IVP テスト・ロギング

fekfivpc.log 189

fekfivpc.log 185

fekfivpi IVP テスト・ロギング

fekfivpi.log 189

fekfivpi.log 185

fekfivpi.log、IVP テスト・ロギング 189

fekfivps.log 185

fekfivps.log、IVP テスト・ロギング 189

FEKLOGS、ログとセットアップの分析に

使用 184

FEKRACF、セキュリティ定義 47

fekrivp 198

ffsget.log 184

ffsput.log 184

ffs.log 184

G

GATE、処分 43

gskkyman による鍵データベースの作成

215

H

Host Connect Emulator 204

I

IEASYSxx 120

MAXUSER 113, 203

ISPF TSO/ISPF クライアント・ゲートウ

エイ

ISP.SISPLOAD 53

ISPF プロファイルの使用、既存の 172

ISPF、複数の割り振り exec の使用 174

ISPF.conf ファイル、複数のセットアップ

での使用 174

ISPF.conf、基本カスタマイズ 172

ISP.SISPLOAD

ISPF TSO/ISPF クライアント・ゲート

ウェイ 53

I/P テスト・ロギング

fekfivpi.log 189

fekfivps.log 189

IVTPRMxx

ECSA MAX 113

FIXED MAX 113

J

Java Xquickstart オプション 126

Java アプリケーションとしての RSE
グラフィカル表現 5

Java 仮想マシン (JVM) 間のクラス共用
126

Java ダンプ 191

Java ヒープ・サイズ、固定 126

Java ヒープ・サイズの限度 99

JAVA_DUMP_TDUMP_PATTERN 191

JCL の要件、始動 201

JES JMON

GEN_CONSOLE_NAME 32

JES コマンド・セキュリティの定義
56

JES ジョブ・モニター (JMON) 4

JES ジョブ・モニター、FEJCNFG 44

JES ジョブ・モニター構成

GEN_CONSOLE_NAME 32

JES ジョブ・モニターのトレース 193

JES ジョブ・モニターの認証 23

JES ジョブ・モニター・ロギング 186

JES セキュリティ 29

JMON 56, 181

JVM 間のクラス共用 126

K

keytool による鍵ストアの作成 217

L

LDAP グループ、開発者の追加 144

LDAP グループの初期セットアップ 143

LDAP グループのセットアップ、初期
143

LDAP サーバーの選択 141

LDAP サーバーのロケーション 142

LDAP スキーマ 140

LDAP の考慮事項 68

LIMIT_COMMANDS 30

LIMIT_VIEW 32

lock.log 184

logon.action、ユーザー出口 168

LPALSTxx 124

M

MVS ダンプ 190

MVS プログラム制御ライブラリーの定
義、RSE の 53

N

netstat 199

O

OFF.REMOTECOPY.MVS 40

OMVS セグメントの定義 50

OutOfMemoryError 203

P

PassTicket サポートの定義、RSE の 54

PassTicket の使用 25

POE 検査 25, 39

Port Of Entry 検査 25, 39

Port Of Entry 検査の定義、RSE 用 39

PORTRANGE 200

pushtoclient.properties 144, 147

R

RACF

許可 59

RACF による鍵リングの作成 207

RESTful インターフェース 152

RESTful インターフェースと Web サービ
ス・インターフェース 152

rmt_class_loader_cache.jar 184

RSE サーバー 66

RSE サーバーでのスレッド・セキュリ
ティー

PassTicket 25

RSE サーバーの定義、セキュアな z/OS
UNIX として 52

RSE スレッド・プールのログ・ファイル
audit.log 186

rsedaemon.log 186

rseserver.log 186

serverlogs.count 186

stderr.*.log 186

stdout.*.log 186

RSE セットアップのクローン作成、既存
の 209

RSE デーモン 66

RSE デーモン (RSED) 4

RSE デーモンと監査ロギング 27

RSE デーモンによる認証 38

RSE デーモンのログ・ファイル

audit.log 186

rsedaemon.log 186

rseserver.log 186

serverlogs.count 186

stderr.*.log 186

stdout.*.log 186

RSE デーモン・ロギング 186

RSE トレース 193

RSE のモニター 114

RSE ユーザー・ロギング

ffsgget.log 187

ffsput.log 187

ffs.log 187

lock.log 187

rmt_class_loader.cache.jar 187

rsecomm.log 187

stderr.log 187

stdout.log 187

.dstoreMemLogging 187

.dstoreTrace 187

rsecomm.log 184

SCLM Developer Toolkit のロギング
188

rsecomm.properties 194

rsedaemon.log 184, 185

rsed.envvars 107, 145, 147, 179

Dmaximum.clients 108

Dmaximum.threadpool.process 108

Dmaximum.threads 108

Dminimum.threadpool.process 108

DSTORE_LOG_DIRECTORY 188, 193

STEPLIB 34

Xms 108

Xmx 108

_CMDSERV_CONF_HOME 174

_RSE_JAVAOPTS 172, 191

_RSE_PORTRANGE 24

rsed.envvars の更新、共存を可能にするた
めの 210

rseserver.log 184, 185

RSE、アプリケーション保護の定義 55

RSE、セキュアな z/OS UNIX サーバーと
して定義 52

RSE、MVS プログラム制御ライブラリー
の定義 53

RSE、PassTicket サポートの定義 54

RSE、Port Of Entry 検査の定義 39

RSE、pushtoclient.properties 47

RSE、rsed.envvars

_RSE_JAVAOPTS 45

RSE、ssl.properties 46

RSE、z/OS UNIX プログラム制御ファイ
ルの定義 62

S

SCLM Developer Toolkit 53
SCLM Developer Toolkit (SCLMDT) 4
SCLM Developer Toolkit のログイン
rsecomm.log 188
SCLM セキュリティー 44
Secure Sockets Layer のセットアップ
205
Secure Sockets Layer ホスト構成接続のテ
スト 211
Secure Sockets Layer を使用した通信暗号
化 24
serverlogs.count 184
SETUID ファイル・システム属性 196
SMP/E インストール、スティッキー・ビ
ット 199
SSL 暗号化通信 43, 155
SSL のセットアップ 205
SSL ホスト構成接続のテスト 211
SSL を使用した通信暗号化 24
SSL、暗号化 206
ssl.properties の更新による SSL のアクテ
ィブ化 210
ssl.properties、新しい RSE デーモンの作
成による SSL のアクティブ化 211
SSL/TLS 暗号化通信 33
stderr.log 184
stderr.*.log 184
stdout.log 184
stdout.*.log 184
STEPLIB の使用の回避 123
SYS1.PARMLIB(BPXPRMxx) 120
MAXASSIZE 52, 201
MAXPROCSYS 203
MAXPROCUSER 203
MAXUIDS 203
SYS1.PARMLIB(BPXPRMxx) で設定され
る制限 201
SYS1.PARMLIB(BPXPRMxx)、Java 仮想
マシン (JVM) 128
SYS1.PARMLIB(IEASYSxx) 120
MAXUSER 203

T

TCP/IP のセットアップ 221
TCP/IP のデフォルト動作、オーバーライ
ド 68
TCP/IP のデフォルト動作のオーバーライ
ド 68
TCP/IP の動作、デフォルトのオーバーラ
イド 68
TCP/IP ポート 65
TCP/IP ポート、グラフィカル表現 65
TCP/IP ポート、予約済み 199

TCP/IP ポートの予約 67
TCP/IP リゾルバー、解決されないホス
ト・アドレス
lock.log 228
TCP/IP、リゾルバーで使用可能なローカル
定義 227
TCP/IP、Developer for System z への適用
225
TLS を使用した通信暗号化 24
TLS、暗号化 206
TSO アクセス方式 171
TSO 環境のカスタマイズ 171
TSO コマンド・サービス 4, 171
TSO/ISPF クライアント・ゲートウェイ・
アクセス方式の使用 172
TSO/ISPF、カスタマイズ -
ISPF.conf 172
TSO/ISPF、既存の ISPF プロファイルの
使用 172
TSO/ISPF、複数のセットアップでの使用
174
TSO/ISPF、複数の割り振り exec の使用
174
TSO/ISPF、割り振り exec の使用 173

U

UNIX 環境で使用される検索順序 223
UNIX サーバーとしての RSE の定義 52
UNIX ダンプ・ロケーション 192
UNIX プログラム制御ファイルの定義、
RSE の 62

V

VIPA、Distributed Dynamic 70

W

Web Owning Region 152
Web サービス・インターフェース 152
WLM に関する考慮事項 xiv, 77
WLM 分類規則 78

X

Xquickstart、Java オプション 126
X.509 クライアント認証サポートの追加
214
X.509 証明書 22
X.509 証明書を使用したクライアント認証
35
x.509 認証、セットアップ 205

Z

zFS ファイル・システムの使用 123
z/OS UNIX REXX exec 167
z/OS UNIX 環境で使用される検索順序
223
z/OS UNIX 許可ビット 196
z/OS UNIX コマンド、便利
chgrp 18
chmod 18
chown 18
ls 18
z/OS UNIX サーバーとしての RSE の定
義 52
z/OS UNIX シェル・スクリプト 166
z/OS UNIX ダンプ・ロケーション 192
z/OS UNIX ディレクトリー構造
グラフィカル表現 15
z/OS UNIX のモニター 115
z/OS UNIX ファイル・システムのモニタ
ー 118
z/OS UNIX ファイル・システム・スパー
スの使用量 105
z/OS UNIX プログラム制御ファイルの定
義、RSE の 62

[特殊文字]

.dstoreMemLogging 184
.dstoreTrace 184
/var/rdz/pushtoclient/*install 145, 147
_RSE_PORTRANGE 24



Printed in Japan

SA88-4226-06



日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21