Network Manager IP Edition

バージョン 3 リリース 9

イベント管理ガイド



R2E2

Network Manager IP Edition

バージョン 3 リリース 9

イベント管理ガイド



R2E2

注記

本書および本書で紹介する製品をご使用になる前に、 289 ページの『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 Tivoli Network Manager IP Edition (5724-S45) のバージョン 3、リリース 9、モディフィケーション 0、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。

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： R2E2
Network Manager IP Edition
Version 3 Release 9
Event Management Guide

発行： 日本アイ・ビー・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

© Copyright IBM Corporation 2006, 2016.

目次

本書について vii
対象読者 vii
本書の内容 vii
資料 ix
アクセシビリティー xii
Tivoli 技術研修 xiii
サポート情報 xiii
この資料の規則 xiv

第 1 章 ネットワークのポーリングについて 1
ポーリング・ポリシー 2
ポーリング・ポリシー・パラメーター 2
ポーリング・ポリシー・スコープ 3
ポーリング定義 4
ポーリング定義パラメーター 5
ポーリング・メカニズム 5
ポーリング定義タイプ 10
データ・ラベル 11
ping ポーリングのプロパティーおよびメトリック 12
ポーリング定義内のマルチバイト・データ 12

第 2 章 ポーリングの有効化と無効化 . . 13

第 3 章 ポーリングの作成 15
フル機能のポーリング・ポリシーの作成 16
単純なポーリング・ポリシーの作成 23
カスタム・データに基づくポーリング・ポリシー作成のためのクイック・リファレンス 24

第 4 章 新規ポーリング定義の作成 . . . 27
基本しきい値ポーリング定義の作成 27
汎用しきい値ポーリング定義の作成 30
シャーシ ping およびインターフェース ping ポーリング定義の作成 32
リモート ping ポーリング定義およびリンク状態ポーリング定義の作成 34

第 5 章 ポーリングの変更 37
ポーリング・ポリシーの変更 37
ポーリング・ポリシーの例 42
ポーリング定義の変更 43
基本しきい値ポーリング定義の変更 44
汎用しきい値ポーリング定義の変更 46
シャーシ ping およびインターフェース ping ポーリング定義の変更 48
リモート ping ポーリング定義およびリンク状態ポーリング定義の変更 50
カスタマイズされたポーリング定義の例 52
基本しきい値の式の例 54
汎用しきい値の式の例 54

第 6 章 ポーリング・ポリシーの削除 . . 57

第 7 章 ポーリング定義の削除 59

第 8 章 適応ポーリングの管理 61
適応ポーリングのシナリオ 61
デバイスが実際にダウンしていることの迅速な確認 61
しきい値違反の迅速な確認 64
適応ポーリングの作成 67

第 9 章 ネットワーク・ポーリングの管理 71
ポーリングの管理 71
SNMP 資格情報のチェックを行わないことによる ncp_poller 始動速度の向上 71
ポーリング状況の取得 72
ポーリングの有効化と無効化 72
ポーリングのリフレッシュ 73
ドメイン間のポーリングのコピー 73
ポーリング中断オプション 74
ポーリング帯域幅の調整 75
リンク状態ポーリングの構成 78
SNMP しきい値ポーリングの構成 79
複数のポーリング・プログラムの管理 79
複数のポーリング・プログラムの概要 79
追加のポーリング・プログラムの設定 81
ポーリング・プログラムの除去 84
ヒストリカル・ポーリング・データの管理 84
ストレージ容量の考慮事項 84
ヒストリカル・ポーリング・データのストレージ限度の増加 88
ヒストリカル・ポーリング・データの削除 . . . 89
ポーリング・プログラムのキャパシティーのモニター 90
エンティティーの状況の照会 95

第 10 章 ping ポーリングのトラブルシューティング 97

第 11 章 イベント・エンリッチおよびイベント相関について 99
イベントの強化 99
イベント・エンリッチのクイック・リファレンス 99
イベント・フィルター 102
イベント状態 110
イベント処理 114

© Copyright IBM Corp. 2006, 2016

例: デフォルトでの Tivoli Netcool/OMNIbus トラップ・イベントの強化 137
イベント・ゲートウェイ・プラグイン 140
プラグインの説明 140
プラグインのサブスクリプション. 152
根本原因の分析. 154
RCA のクイック・リファレンス 155
優先順位値 156
ポーリング・プログラム・エンティティー. . . 158
RCA および非管理状況 159
RCA スティッチャー. 161
根本原因分析の例 166
RCA によって使用されるトポロジー・パスの検査 175

第 12 章 イベント・エンリッチの構成 181
追加のイベント・エンリッチの構成 181
ObjectServer alerts.status テーブルに対する変更 181
例: メイン・ノード・デバイスのロケーションによるイベントの強化 182
例: インターフェース名によるイベントの強化 184
ObjectServer 更新間隔フィールドの構成 185
OQL サービス・プロバイダーを使用してイベント・ゲートウェイ・データベースにログインする . 187
ObjectServer の照会 187
NCIM データベースの照会. 187
イベントと ObjectServer の再同期化 188
共通イベント・ゲートウェイ・プロパティーの構成 189

第 13 章 イベント・ゲートウェイ・プラグインの構成 191
プラグインの有効化と無効化 191
プラグイン情報のリスト. 192
イベント・マップ・サブスクリプションの変更 . . 193
プラグイン構成パラメーターの設定 195
SAE プラグインの構成 197
サービスに影響を与えるイベントの要約フィールド情報の構成 197
SAE プラグインへの SAE タイプの追加 . . . 198

第 14 章 根本原因分析の構成 201
ポーリング・プログラム・エンティティーの構成 201
イベントの存続期間の最大差の構成 202
クロスドメイン・ネットワークでの RCA の考慮事項 203

付録 A. デフォルトのポーリング・ポリシー 205
デフォルトの ping ポリシー 205
デフォルトのリモート ping ポリシー 206
デフォルトの SNMP しきい値ポリシー 206
デフォルトの SNMP リンク状態ポリシー. . . . 210
レポート作成で使用されるポーリング・ポリシー 210

付録 B. デフォルトのポーリング定義 211

付録 C. トリガーしきい値とクリアのしきい値の例. 219

付録 D. ポーリング定義の式の構文. . . 221
しきい値の式での eval ステートメントの構文 . . 221
SNMP 変数に対する eval ステートメントの構文 221
ネットワーク・エンティティー変数に対する eval ステートメントの構文. 222
ポーリング・ポリシー変数に対する eval ステートメントの構文. 223
ポーリング定義変数に対する eval ステートメントの構文 224
しきい値の式での演算子. 225

付録 E. Tivoli Netcool/OMNIbus のプローブ の構成 227
nco_p_ncpmonitor.props ファイルについて . . . 227
nco_p_ncpmonitor.rules ファイルについて . . . 228
nco_p_ncpmonitor.rules 構成のリファレンス. . . 228
ルール・ファイル処理の例 229
Network Manager イベント・データ・フィールド 231
Network Manager で使用する alerts.status フィールド 234

付録 F. Network Manager のイベント・カテゴリー 241
Network Manager ネットワーク・イベント . . . 242
Network Manager 状況イベント. 242

付録 G. ポーリング・データベース. . . 249
NCMONITOR データベース 249
ncmonitor データベース内のポーリング用の SNMP テーブル 249
ping ポーリング状況テーブル. 252
OQL データベース (OQL databases) 262
ポーリング用の構成データベース. 262
ポーリング用のプロファイル・データベース . . 266

付録 H. イベントの強化のデータベース 269
ncp_g_event データベース. 269
構成データベース・スキーマ 269
ncp_g_event プラグイン・データベース 276
RCA プラグイン・データベース 276
SAE プラグイン・データベース 279
ncmonitor 内の ncp_g_event プラグイン・データベース表 280

付録 I. Network Manager 用語集 . . . 283

特記事項. 289
商標 291

索引 **293**

本書について

IBM Tivoli Network Manager IP Editionには、ネットワーク・ディスカバリー、デバイス・モニター、トポロジー可視化、および根本原因分析 (RCA) の詳細機能があります。Network Manager は、さまざまなネットワークを管理するために、広範囲にわたってカスタマイズおよび構成することができます。また、Network Manager には豊富なレポート作成機能があり、さらに IBM Tivoli Application Dependency Discovery Manager、IBM Tivoli Business Service Manager、IBM Systems Director などの他の IBM 製品との統合も実現します。

「*IBM Tivoli Network Manager IP Edition* イベント管理ガイド」では、IBM® Tivoli® Network Manager IP Edition を使用して、ネットワーク・デバイスをポーリングする方法について説明します。

対象読者

本書は、IBM Tivoli Network Manager IP Edition の構成を行うユーザー、システム管理者、およびネットワーク管理者を対象としています。

IBM Tivoli Network Manager IP Edition は、IBM Tivoli Netcool/OMNIbus と連携して動作します。この資料は、IBM Tivoli Netcool/OMNIbus の機能を把握していることを前提としています。 IBM Tivoli Netcool/OMNIbus について詳しくは、 ix ページの『資料』に記載されている資料を参照してください。

本書の内容

本書には、以下のセクションがあります。

- 1 ページの『第 1 章 ネットワークのポーリングについて』

  ポーリング・ポリシーとポーリング定義、およびそれらがどのように相互作用してネットワーク・ポーリングを作成するかについて説明します。
- 13 ページの『第 2 章 ポーリングの有効化と無効化』

  ポーリングを有効にしたり無効にする方法について説明します。
- 15 ページの『第 3 章 ポーリングの作成』

  既存のポーリングのコピーをし、かつポーリング・ポリシー・ウィザードを使用してポーリングを作成する方法について説明します。
- 27 ページの『第 4 章 新規ポーリング定義の作成』

  新規ポーリング定義の作成方法について説明します。
- 37 ページの『第 5 章 ポーリングの変更』

  ポーリングの変更方法について説明します。
- 57 ページの『第 6 章 ポーリング・ポリシーの削除』

© Copyright IBM Corp. 2006, 2016

ポーリング・ポリシーが必要でなくなった場合の削除方法について説明します。

- 59 ページの『第 7 章 ポーリング定義の削除』

ポーリング定義が必要でなくなった場合の削除方法について説明します。

- 61 ページの『第 8 章 適応ポーリングの管理』

適応ポーリングは、ネットワークのイベントに動的に反応します。この章では、さまざまなネットワーク問題のシナリオを管理する適応ポーリングについて説明します。

- 71 ページの『第 9 章 ネットワーク・ポーリングの管理』

コマンド行インターフェースを使用して、複数のポーリング・プログラムの管理、ネットワーク・ドメイン間のネットワーク・ポーリングのコピー、ネットワーク・ポーリングの中断を行う方法について説明します。

- 97 ページの『第 10 章 ping ポーリングのトラブルシューティング』

ネットワーク内の重要な IP アドレスが Network Manager によって期待どおりにポーリングされているかどうかを確認する方法について説明します。

- 99 ページの『第 11 章 イベント・エンリッチおよびイベント相関について』

イベント・ゲートウェイによるイベント・エンリッチの実行方法、およびイベントが根本原因分析 (RCA) やフェイルオーバーなどのプラグイン・プロセスに渡され、強化されたイベントに含まれているデータに基づいてさらなるアクションが行われる方法について説明します。また、強化されたイベントが ObjectServer に戻されるメカニズムについても説明します。

- 181 ページの『第 12 章 イベント・エンリッチの構成』

イベント・ゲートウェイを通過するときのイベント処理の構成方法について説明します。

- 191 ページの『第 13 章 イベント・ゲートウェイ・プラグインの構成』

イベント・ゲートウェイ・プラグインの構成方法について説明します。

- 201 ページの『第 14 章 根本原因分析の構成』

イベント・ゲートウェイ RCA プラグインの構成方法について説明します。

- 205 ページの『付録 A. デフォルトのポーリング・ポリシー』

IBM Tivoli Network Manager IP Edition のインストールに含まれているポーリング・ポリシーについて説明します。

- 211 ページの『付録 B. デフォルトのポーリング定義』

IBM Tivoli Network Manager IP Edition のインストールに含まれているポーリング定義について説明します。

- 219 ページの『付録 C. トリガーしきい値とクリアのしきい値の例』

汎用しきい値ポーリング定義のクリアのしきい値とトリガーしきい値を設定するための、しきい値の数式を例示します。

- 221 ページの『付録 D. ポーリング定義の式の構文』

基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成するときに役立つ参照情報です。

- 227 ページの『付録 E. Tivoli Netcool/OMNIbus のプローブ の構成』

  Network Manager IP Edition ポーリングにより生成されるイベントを Tivoli Netcool/OMNIbus ObjectServer に送信できるようにするプローブ、Tivoli Netcool/OMNIbus のプローブについて説明します。

- 241 ページの『付録 F. Network Manager のイベント・カテゴリー』

  Network Manager によって生成されるイベントは、2 つのカテゴリーに分類されます。1 つはモニターされているネットワークに関するイベント、もう 1 つは Network Manager のプロセスに関するイベントです。この付録には、これらのイベントに関する詳細情報が記載されています。

- 249 ページの『付録 G. ポーリング・データベース』

  ポーリングに使用されるデータベースの構造について説明します。

- 269 ページの『付録 H. イベントの強化のデータベース』

  イベント・エンリッチに使用されるデータベースの構造について説明します。

資料

このセクションでは、Network Manager ライブラリーの資料と関連資料を紹介します。また、このセクションでは Tivoli の資料にオンラインでアクセスする方法と、Tivoli 資料のご注文方法も説明します。

Network Manager ライブラリー

Network Manager ライブラリーに収録されている資料を以下に示します。

- *IBM Tivoli Network Manager IP Edition* リリース情報, GI88-4254

  IBM Tivoli Network Manager IP Edition に関する重要情報および最新情報を提供します。この資料は、デプロイメント担当者および管理者を対象としており、最初に確認する必要があります。

- *IBM Tivoli Network Manager* スタートアップ・ガイド, GI11-9353-00

  製品のインストール後に、IBM Tivoli Network Manager IP Edition をセットアップする方法について説明します。このガイドは、製品を開始する方法、製品が正しく実行されていることを確認する方法、およびネットワークをディスカバーする方法について説明します。良好なネットワーク・ディスカバリーを取得することは、Network Manager IP Edition を正しく使用するために重要です。このガイドは、最初のディスカバリーを構成およびモニターする方法、ディスカバリーの結果を検証する方法、実動ディスカバリーを構成する方法、およびネットワーク・トポロジーを最新の状態に保つ方法について説明します。ネットワーク・トポロジーを最新の状態にした後に、このガイドで、ネットワーク・トポロジーをネットワーク・オペレーターで使用可能にする方法、およびネットワークをモニターする方法について確認してください。この簡潔なガイドでは、重要なタスクについて取り上げ、より詳細なタスク、オプションのタスク、または拡張タスクに言及し、残りの資料セット内の参照資料を掲載しています。

- *IBM Tivoli Network Manager IP Edition* 製品概要, GC27-2759-00

  IBM Tivoli Network Manager IP Edition の概要について説明します。製品体系、コンポーネント、および機能について説明します。この資料は、IBM Tivoli Network Manager IP Edition に関心のあるユーザーを対象としています。
- *IBM Tivoli Network Manager IP Edition* インストールと構成ガイド , SC27-2760-00

  IBM Tivoli Network Manager IP Edition のインストール方法について説明します。また、必須およびオプションのインストール後の構成タスクについても説明します。この資料は、IBM Tivoli Network Manager IP Edition のインストールおよびセットアップを行う必要のある管理者を対象としています。
- *IBM Tivoli Network Manager IP Edition* 管理ガイド, SC27-2761-00

  プロセスの管理、データベースの照会、および製品の開始と停止の方法など、IBM Tivoli Network Manager IP Editionの管理タスクについて説明しています。この資料は、IBM Tivoli Network Manager IP Edition の保守および可用性を担当する管理者を対象としています。
- *IBM Tivoli Network Manager IP Edition* ディスカバリー・ガイド, SC27-2762-00

  IBM Tivoli Network Manager IP Edition を使用してネットワークを検出する方法について説明します。この資料は、ネットワーク・ディスカバリーの構成および実行を担当する管理者を対象としています。
- *IBM Tivoli Network Manager IP Edition* イベント管理ガイド, SC27-2763-00

  IBM Tivoli Network Manager IP Edition を使用して、ネットワーク・デバイスのポーリング、ネットワーク・デバイスからの各種イベントの構成、および Tivoli Netcool/OMNIbus イベント・ゲートウェイに対するプラグインの管理 (根本原因分析のための RCA プラグインの構成など) を行う方法について説明しています。この資料は、ネットワーク・ポーリング、イベントの強化、根本原因分析、およびイベント・ゲートウェイ・プラグインの構成と実行を担当する管理者を対象としています。
- *IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイド, GC27-2765-00

  IBM Tivoli Network Manager IP Edition を使用して、製品によって識別されるネットワーク問題を解決する方法について説明します。この資料は、ネットワークの問題を識別または解決することを担当するネットワーク・オペレーターを対象としています。
- *IBM Tivoli Network Manager IP Edition* ネットワーク可視化セットアップ・ガイド, SC27-2764-00

  IBM Tivoli Network Manager IP Edition ネットワーク可視化ツールを構成して、カスタマイズされた作業環境をネットワーク・オペレーターに提供する方法について説明しています。この資料は、ネットワーク・オペレーターの作業を支援する役割を担う製品管理者またはチーム・リーダーを対象としています。
- *IBM Tivoli Network Manager IP Edition* データベース管理ガイド, SC27-2767-00

IBM Tivoli Network Manager IP Edition でのコンポーネント・データベースのスキーマについて説明します。 この資料は、コンポーネント・データベースを直接照会する必要のある上級者を対象としています。

- *IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンス, SC27-2766-00

  トポロジー・データを IBM Tivoli Network Manager IP Edition に保管するために使用するデータベースのスキーマについて説明します。この資料は、トポロジー・データベースを直接照会する必要のある上級者を対象としています。

- *IBM Tivoli Network Manager IP Edition* 言語リファレンス , SC27-2768-00

  IBM Tivoli Network Manager IP Editionで使用されるシステム言語 (Stitcher 言語、オブジェクト照会言語など) について説明します。この資料は、IBM Tivoli Network Manager IP Edition の操作をカスタマイズする必要のある上級者を対象としています。

- *IBM Tivoli Network Manager IP Edition Perl API* ガイド, SC27-2769-00

  開発者が、IBM Tivoli Network Manager IP Edition と対話するカスタム・アプリケーションを作成するために使用できる Perl モジュールについて説明しています。開発者が作成できるカスタム・アプリケーションの例としては、ポーリング・エージェントやディスカバリー・エージェントなどがあります。この資料は、そのようなカスタム・アプリケーションを作成する必要がある上級 Perl 開発者を対象としています。

- *IBM Tivoli Monitoring for Tivoli Network Manager IP Edition* ユーザー・ガイド, SC27-2770-00

  IBM Tivoli Monitoring for IBM Tivoli Network Manager IP Edition のインストールおよび使用に関する情報が記載されています。この資料の対象読者は、IBM Tivoli Monitoring for IBM Tivoli Network Manager IP Edition をインストールおよび使用して、IBM Tivoli Network Manager IP Edition リソースをモニターおよび管理するシステム管理者です。

前提資料

この資料の情報を有効に活用するために必要な前提知識を得るには、以下の資料を参照してください。

- *IBM Tivoli Netcool/OMNIbus* インストールおよびデプロイメント・ガイド , SC88-8220-00

  Tivoli Netcool/OMNIbus のインストールおよびアップグレード手順を含み、セキュリティーおよびコンポーネント通信の構成方法を説明します。また、Tivoli Netcool/OMNIbus アーキテクチャーの例が含まれており、その実装方法について説明します。

- *IBM Tivoli Netcool/OMNIbus* ユーザーズ・ガイド, SC88-8226

  デスクトップ・ツールについて概説し、これらのツールを使用したイベント管理に関連したオペレーター・タスクについて説明します。

- *IBM Tivoli Netcool/OMNIbus* 管理ガイド, SC88-8221

Tivoli Netcool/OMNIbus 管理者 GUI、コマンド行ツール、およびプロセス制御を使用して管理用タスクを実行する方法を説明します。また、ObjectServer SQL の構文および自動化の説明と例も含まれています。

- *IBM Tivoli Netcool/OMNIbus* プローブとゲートウェイ・ガイド, SC88-8223

  プローブ・ルール・ファイル構文やゲートウェイ・コマンドなどの、プローブおよびゲートウェイに関する概説および参照情報が含まれています。

- *IBM Tivoli Netcool/OMNIbus Web GUI* 管理およびユーザーズ・ガイド SC88-8222

  Tivoli Netcool/OMNIbus Web GUI を使用した、管理タスクおよびイベント視覚化タスクの実行方法が記載されています。

用語集へのオンライン・アクセス

以下は英語のみの対応となります。IBM Terminology Web サイトには、IBM product ライブラリーの用語が 1 つにまとめられています。Terminology Web サイトの Web アドレスは以下のとおりです。

http://www.ibm.com/software/globalization/terminology

マニュアルへのオンライン・アクセス

以下は英語のみの対応となります。IBM では、この製品およびその他のすべての Tivoli 製品に関する資料を、使用可能になった時点および更新された時点で、IBM Knowledge Center の Web サイトに載せています。

http://www-01.ibm.com/support/knowledgecenter/

Network Manager 資料は、その Web サイトの「**Cloud & Smarter Infrastructure**」ノードの下にあります。

注**:** PDF 文書をレターサイズ以外の用紙に印刷する場合は、PDF 読み取りアプリケーションのメニューから「ファイル」 > 「印刷」を選択して表示されたウィンドウでオプションを設定し、レターサイズのページをご使用の用紙に印刷できるようにしてください。

アクセシビリティー

アクセシビリティー機能は、運動障害または視覚障害など身体に障害を持つユーザーがソフトウェア・プロダクトを快適に使用できるようにサポートします。

アクセシビリティー機能

以下のリストは、Network Manager での主なアクセシビリティー機能です。

- キーボードのみで操作可能なコンソール・ベースのインストーラー
- 画面読み上げ機能がサポートされたコンソール・ベースのインストーラー
- Network Manager には、弱視の方に適した以下の機能が用意されています。
  - GUI で使用されている、テキスト以外のすべてのコンテンツには、代替テキストが関連付けられています。

- 弱視の方は、高コントラスト・モードなどのシステム画面設定を調整したり、ブラウザー設定を使用してフォント・サイズを制御したりできます。
- 情報の伝達、アクションの指示、対応の要求、または資格要素の識別を行ううえで、色以外の視覚的手段も使用しています。

- Network Manager には、光過敏性てんかんを患う方に適した以下のような機能があります。
  - Web ページには、1 秒間に 2 回を超える明滅を起こすものは含まれていません。

Network Manager Knowledge Center のアクセシビリティーは、その Knowledge Center 内に説明されています。

アクセシビリティー機能を使用できるように Internet Explorer を構成するための追加ステップ

Web ブラウザーとして Internet Explorer を使用している場合は、追加の構成ステップを実行し、アクセシビリティー機能を使用可能にすることをお勧めします。

高コントラスト・モードを使用可能にするには、以下の手順を実行します。

1. 「ツール」 > 「インターネット オプション」 > 「ユーザー補助」をクリックします。
2. 「書式設定」セクションのすべてのチェック・ボックスを選択します。

「表示」 > 「文字のサイズ」 > 「最大」をクリックしてもフォント・サイズが大きくならない場合は、**Ctrl** キーと **+** キーおよび **Ctrl** キーと **-** キーを使用してください。

IBM とアクセシビリティー

アクセシビリティーに対する IBM のコミットメントについては、「IBM Human Ability and Accessibility Center」を参照してください。

Tivoli 技術研修

以下は英語のみの対応となります。 Tivoli 技術研修の情報については、以下の IBM Tivoli Education Web サイトを参照してください。

http://www.ibm.com/software/tivoli/education

サポート情報

ご使用の IBM ソフトウェアに問題がある場合、迅速な解決が必要となります。IBM では、お客様が以下の方法で必要なサポートを受けることができるようにしています。

オンライン

IBM ソフトウェア・サポート サイト (http://www.ibm.com/software/support/probsub.html) にアクセスし、指示に従ってください。

IBM Support Assistant

IBM Support Assistant (ISA) は、無料のローカル・ソフトウェア保守性ワークベンチで、IBM ソフトウェア製品での疑問や問題の解決に役立ちます。ISA を使用すると、サポート関連情報と問題判別のための保守性ツールに素早くアクセスできます。ISA ソフトウェアをインストールするには、http://www.ibm.com/software/support/isa にアクセスしてください。

この資料の規則

この資料では、特殊な用語およびアクションと、オペレーティング・システムに依存するコマンドおよびパスに、いくつかの規則を使用しています。

書体の規則

この資料では、以下のような書体の規則を使用しています。

太字

- 太字にしなければ周囲の本文と区別しにくい小文字のコマンドおよび大/小文字混合のコマンド
- インターフェース・コントロール (チェック・ボックス、プッシュボタン、ラジオ・ボタン、スピン・ボタン、フィールド、フォルダー、アイコン、リスト・ボックス、リスト・ボックス内の項目、複数列のリスト、コンテナー、メニュー選択、メニュー名、タブ、プロパティー・シート)、ラベル (「ヒント:」および「オペレーティング・システムの考慮事項:」など)
- 本文中のキーワードおよびパラメーター

斜体

- 引用 (例: 資料、ディスケット、および CD のタイトル)
- 本文中で定義された単語 (例: 非交換回線は *Point-to-Point* 回線と呼ばれます)
- 単語および文字の強調 (単語自体を強調する例: 「*that* という単語は、制限節を導入します。」、文字自体を強調する例: 「LUN アドレスは *L* という文字で始まっていなければなりません。」)
- 本文での新規用語 (定義リスト内を除く) (例: ビュー とは、データを含むワークスペース内のフレームです)
- 指定する必要のある変数および値 (例: ... ここで *myname* は .... を表します)

モノスペース

- 例およびコード例
- ファイル名、プログラミングのキーワード、および周囲の本文から区別しにくいその他のエレメント
- ユーザーに対して出されるメッセージ・テキストおよびプロンプト
- ユーザーが入力する必要のあるテキスト
- 引数またはコマンド・オプションの値

オペレーティング・システムに依存する変数およびパス

本書では、コマンドが特定のプラットフォームを対象とする場合を除き、プラットフォーム固有のプレフィックスおよびサフィックスを付けずに環境変数を記載しています。例えば、Network Manager コア・コンポーネントがインストールされているディレクトリーを、NCHOME として表します。

Windows コマンド行を使用する場合、環境変数にはパーセント記号 **%** をプレフィックスおよびサフィックスとして付加し、ディレクトリー・パスのスラッシュ (**/**) はそれぞれ円記号 (**¥**) に置き換えてください。例えば、Windows システムでは、NCHOME は %NCHOME% です。

UNIX システムの場合、環境変数にはドル記号 **$** をプレフィックスとして付加してください。例えば、UNIX では、NCHOME は $NCHOME です。

環境変数の名前は、Windows 環境と UNIX 環境で同じであるとは限りません。例えば、Windows 環境の %TEMP% は、UNIX 環境の $TMPDIR に相当します。Windows システムで bash シェルを使用している場合は、UNIX の規則を使用できます。

第 1 章 ネットワークのポーリングについて

ネットワークをポーリングするために、Network Manager は定期的にネットワーク上のデバイスに照会を送信します。これらの照会によって、例えばデバイスの操作状況や情報管理ベース (MIB) 変数のデータなど、デバイスの動作を判別します。

ネットワーク・ポーリングは、ポーリング・ポリシーによって制御されます。ポーリング・ポリシーは、以下で構成されます。

- ポーリング定義。取得するデータを定義します。
- ポーリング・スコープ。ポーリングするデバイスで構成されます。スコープは、ポーリング定義レベルで変更し、デバイス・クラスおよびインターフェースに基づいてフィルタリングすることもできます。
- ポーリング間隔およびその他のポーリング・プロパティー。

Network Managerは、ネットワークをモニターするために、IBM Tivoli Netcool/OMNIbusSNMP トラップ・プローブおよび Syslog プローブを使用します。Tivoli Netcool/OMNIbus プローブを実行するには、Tivoli Netcool/OMNIbus プロセス制御を使用します。

Tivoli Netcool/OMNIbus プロセス制御の使用法の詳細については、*IBM Tivoli Netcool/OMNIbus* 管理ガイドを参照してください。

ポーリング・プロセスは ncp_poller プロセスによって制御されます。 ncp_poller プロセスは、SNMP 情報を ncmonitor データベースに保管します。他のデータはメモリー内に保管されます。

Network Manager は、ロードを分散させるための複数のポーリング・プログラム・メカニズムを持っています。デフォルトのポーリング・プログラムがネットワークのポーリング要求に対処できない場合、複数のポーリング・プログラム機能の使用が必要な場合があります。

関連タスク:

79 ページの『複数のポーリング・プログラムの管理』
ネットワークをポーリングするために複数のポーリング・プログラムが必要な場合は、Network Manager を設定して複数のポーリング・プログラム機能を管理することができます。ポーリング・プログラムの追加や削除、およびポーリング・プログラム ID を使用して特定のポーリング・プログラムにポリシーを関連付けることができます。

関連資料:

249 ページの『ncmonitor データベース内のポーリング用の SNMP テーブル』
ncmonitor データベース内の SNMP テーブルは、ポーリング・エンジン (ncp_poller) によって使用され、SNMP を使用してディスカバーされた各デバイスへのアクセス方法に関する情報を保管します。

© Copyright IBM Corp. 2006, 2016

ポーリング・ポリシー

ポーリング・ポリシーは、ネットワーク・ポーリング操作のプロパティーすべてを含んでいます。ポーリング・ポリシーでは、デバイスのポーリング頻度、ポーリング実行のために採用するポーリング・メカニズムのタイプ、およびポーリングするデバイスを指定します。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』

Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

ポーリング・ポリシー・パラメーター

ここでは、ポーリング・ポリシーのパラメーターについて説明します。

ポーリング・ポリシーを使用して、以下のパラメーターを定義します。

- ポーリング・ポリシーの名前
- 有効化または無効化: ポーリングが行われるためには、ポーリング・ポリシーを有効化しなければなりません。
- ポーリング定義: 1 つのポーリング・ポリシーに 1 つ以上のポーリング定義を関連付けることができます。インターフェース・レベル・フィルターが必要な場合、ポーリング定義に特定の設定を含めることが求められます。ポリシーに関連付けられたポーリング定義ごとに、ヒストリカル・レポート用のポーリング・データを保管するかどうかを指定できます。このパラメーターが設定されている場合、そのデータは ncpolldata データベース・スキーマに保管されます。

  制約事項: Cisco リモート ping、Juniper リモート ping、および汎用しきい値の各ポーリング定義では、ポーリング・データの保管はサポートされません。
- ポーリング間隔
- 複数のポーリング・プログラム機能がセットアップされる場合に、ポーリング・ポリシーを割り当てるポーリング・プログラム。
- スコープ。これには以下のものが含まれます。
  - ネットワーク・ビュー: ポーリングするデバイスを含むネットワーク・ビューを指定します。
  - デバイス・フィルター: Network Connectivity and Inventory Model (NCIM) データベースの `mainNodeDetails` テーブル内のフィールド値と照らしてフィルタリングすることにより、ポーリングするデバイスのリストを絞り込みます。ブール関係によって複数のフィルターを結合できます。

Network Manager IP Editionは、デフォルトのポーリング・ポリシーおよび定義を提供しています。Network Manager IP Editionのインストール・プロセスの際にポーリング設定をマイグレーションした場合は、他のポーリングも使用可能な場合があります。

ポーリング・ポリシー・スコープ

ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

ポーリング・ポリシー・スコープは、一連のフィルターとして記述できます。どのステージにおいてもフィルターが定義されていない場合は、すべてのデバイスがそのステージを通過します。この一連のフィルターの出力は、一連のデバイスにすることもできますし、(インターフェース・フィルターが定義されている場合は) 一連のデバイス・インターフェースにすることもできます。これを以下の図で説明します。

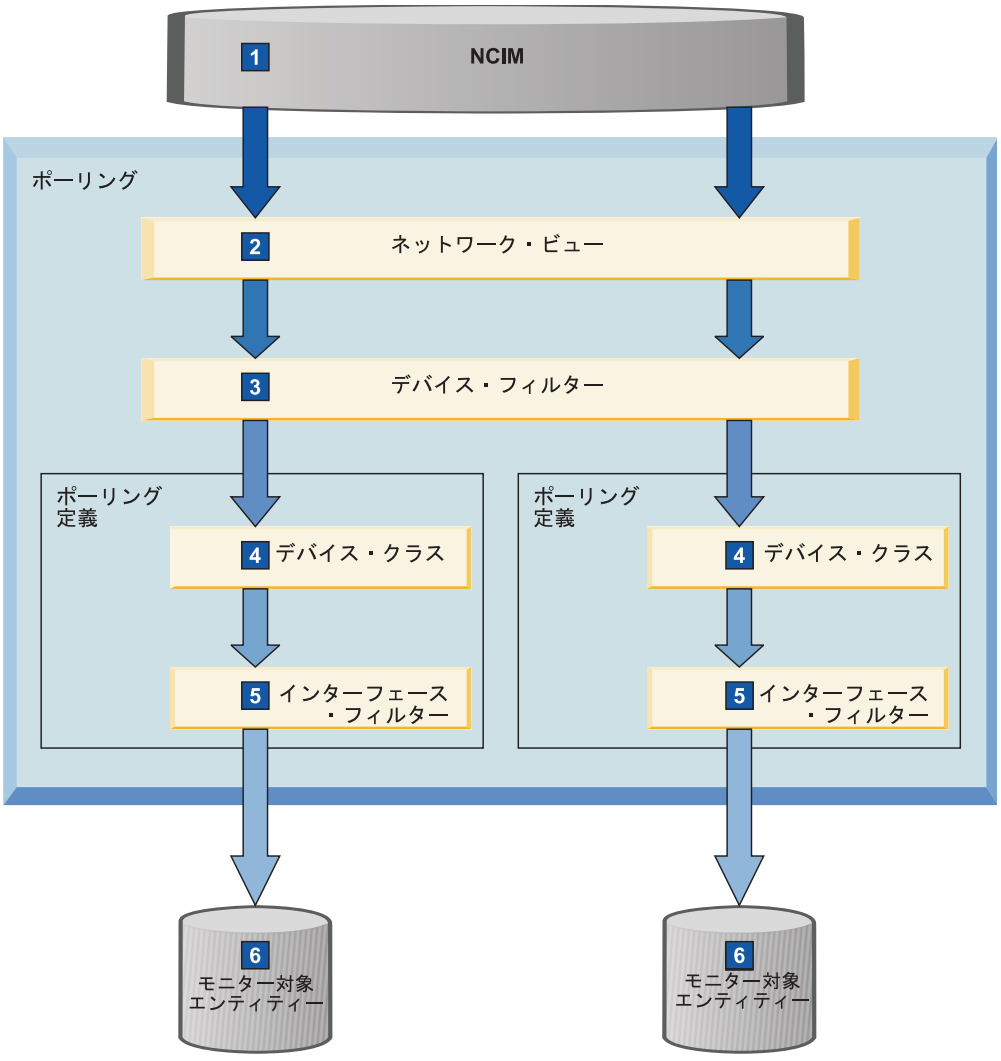


図 *1.* ポーリング・ポリシー・スコープ

1 **NCIM**

NCIM トポロジー・データベースの単一のドメインに定義されたすべてのデバイスから開始します。

2 ネットワーク・ビュー

いずれかのネットワーク・ビューがポーリング・ポリシーに関連付けられている場合、ポリシー・スコープは、それらのビューに含まれるデバイスに制

限されます。ネットワーク・ビューがポーリング・ポリシーに関連付けられていない場合は、すべてのデバイスがこのステージを通過します。この 2 番目の状況は、ポーリング・ポリシー・エディターとポーリング・ポリシー・ウィザードの「ネットワーク・ビュー」タブで「すべてのデバイス」オプションを選択することと同じです。

3 デバイス・フィルター
ポーリング・ポリシーにデバイス・フィルターが定義されている場合、ポリシー・スコープは、そのフィルターに一致する一連のデバイスへとさらに制限されます。デバイス・フィルターが定義されていない場合は、ネットワーク・ビュー・フィルターを通過したすべてのデバイスがこのステージを通過します。この時点で、このポーリング・ポリシーのスコープ内にある一連のデバイスが使用可能になります。ポリシーに割り当てられたポーリング定義ごとに、詳細なフィルタリングに基づいて、さまざまな一連のネットワーク・エンティティーがスコープ内に存在することが可能です。

4 デバイス・クラス
ポリシーに割り当てられたポーリング定義ごとに、デバイス・クラスは、クラスの選択に基づいて、スコープ内のデバイスを制限します。デバイス・クラスが選択されていない場合、フィルタリングは実行されません。

5 インターフェース・フィルター
インターフェース・フィルターが定義されている場合、このインターフェース・フィルターが問題のポーリングで有効であるとすると、このインターフェース・フィルターは、上記のフィルターを通過したデバイスに含まれるすべてのインターフェースに適用されます。出力は、一連のスコープ内インターフェースになります。インターフェース・フィルターが定義されていない場合、出力は、デバイス・クラス・フィルターを通過した一連のデバイスになります。

ポーリング定義

ポーリング定義は、ネットワーク・エンティティーをどのようにポーリングするかを決定します。各ポーリング・ポリシーを、少なくとも 1 つのポーリング定義に関連付ける必要があります。1 つのポーリング・ポリシーを複数のポーリング定義に関連付けることができます。

関連資料:

211 ページの『付録 B. デフォルトのポーリング定義』

Network Manager IP Editionには、一般的なポーリング要件に対応したさまざまなデフォルトのポーリング定義が用意されています。

ポーリング定義パラメーター

ここでは、ポーリング定義のパラメーターについて説明します。

Network Managerは、デフォルトのポーリング・ポリシーおよび定義を提供しています。

ポーリング定義を使用して、以下のパラメーターを定義します。

- ポーリング定義名
- ポーリング定義タイプ: これは、ポーリング定義が使用する「ポーリング・メカニズム」を決定します。以下のポーリング・メカニズムが使用されます。
  - ping ポーリング
  - SNMP ポーリング
- 生成されるイベントの重大度レベル

  重要**:** 重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。 使用可能な重大度レベルのリストについては、「*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイド」を参照してください。
- ポーリング定義の簡略説明
- 基本しきい値ポーリング定義タイプのみ: データ・ラベル。基本しきい値ポーリング定義で収集されたデータをレポートに関連付けるために使用するラベル。レポートを定義するときは、データ・ラベルを使用して、レポートに表示するデータを選択します。これにより、同じデータ・ラベルでタグ付けされているが、複数のポーリング定義で収集されたデータをレポートに表示できます。要約レポートのリストについては、*IBM Tivoli Network Manager IP Edition* 管理ガイドを参照してください。
- ポーリング定義のスコープ: 適用するデバイス・クラスとインターフェース・フィルター (オプション)
- しきい値ポーリング定義のみ: アラートを生成およびクリアするためのしきい値の設定

ポーリング・メカニズム

ポーリング定義は、ping ポーリングおよび SNMP ポーリングという、2 つの可能なポーリング・メカニズムのいずれかを使用します。すべてのポーリング定義は、これらのメカニズムのいずれかに基づいています。

ping ポーリング

ping ポーリングは、ICMP エコー要求を使用して、ネットワーク・デバイスまたはインターフェースの可用性を判別します。

ping プロセスは定期的に ICMP パケットを IP アドレスに送信して応答を待つことにより、デバイスが依然として存在し、生きていて、ネットワーク中でアクセス可能であることを保証します。

ping ポーリングによって以下の結果が生じ得ます。

正常に完了
: ping パケットに対する応答が受信されました。アラートは生成されません。

失敗
: ポーリング定義で指定された時間内に、ping パケットに対する応答が受信されません。応答しないネットワーク・エンティティーについてのアラートが発生します。

復元
: 最後の ping 試行でアクセス不能であったデバイスが、再びアクセス可能になりました。ping 障害アラートをクリアするようにという、アラートが生成されます。

ping ポーリングは、デバイスのシャーシ上またはインターフェース上のいずれかで実行できます。シャーシの場合、ICMP パケットはメイン・ノード・デバイスの IP アドレスに送信されます。メイン・ノード IP アドレスは、インターフェースにも関連付けられています。インターフェースの場合、ICMP パケットは各インターフェースの IP アドレスに送信されます。従って、シャーシとインターフェースの両方について ping ポーリングを有効にすると、メイン・ノード IP アドレス上のトラフィックは 2 倍になります。

要確認**:** デフォルトでは、ディスカバーされたネットワーク・トポロジー内のすべてのデバイス上で有効になっているのはシャーシ ping ポーリングだけです。ただし、デスクトップやプリンターなどのエンド・ノード・デバイスは例外です。

SNMP ポーリング

SNMP ポーリングでは、動作不良または接続障害の原因を判別するために、デバイスから管理情報ベース (MIB) の変数を取得する必要があります。次に、定義済みの数式を、取り出した MIB 変数に適用することにより、障害のあるデバイスまたは接続が診断されます。

SNMP ヘルパーは、SNMP 要求をネットワーク・デバイスに発行するためにポーリングで使用されます。SNMP ヘルパーの構成方法については、「*IBM Tivoli Network Manager IP Edition* インストールと構成ガイド 」を参照してください。

リンク状態ポーリング**:**

リンク状態ポーリングは、ifOperStatus および ifAdminStatus インターフェース MIB 変数における状況の変化をモニターします。これら変数の値がポーリング間隔の間に変化すると、イベントが発生します。

Fix Pack 4 リンク状態ポーリングの初期状態は、以下のように判別できます。

- インターフェース上の既存のリンク状態イベントの状態を確認します。インターフェースに既存のリンク状態イベントがない場合は、初期状態は「クリア」に設定されます。
- `$NCHOME/etc/precision/NcPollerSchema.cfg` 構成ファイル内の **`UseFirstPollForInitialState`** パラメーターを設定して、インターフェースをポーリングします。

例

ifOperStatus の値が以前のポーリングの際に 1 (up) で、今回のポーリングで 2 (down) に変わった場合、イベントが発生します。

次の表は、インターフェース状況の変化によって生成されるイベントを示しています。また、ポーリングがデータを返すことに失敗した場合はイベントが生成されます。その後、同一のデバイスへのポーリングが成功した場合は、明確な重大度を備えたイベントが生成されます。

表 *1. SNMP* リンク状態ポーリングによって生成されるイベント

ポーリング間隔の間での **ifAdminStatus MIB** 変数の状況	ポーリング間隔の間での **ifOperStatus MIB** 変数の状況	生成されるイベント	イベント重大度
1 (up) のまま	1 (up) から 2 (down) に変化	インターフェースがダウンしました。(The interface has gone down.)	軽微
1 (up) のまま	2 (down) から 1 (up) に変化	インターフェースがアップしました。(The interface has come up.)	クリア
1 (up) から 2 (down) に変化	2 (down) から 1 (up) に変化	インターフェースをダウンするべきですが、アップされました。(The interface has come up, although it should be down.)	クリア
1 (up) から 2 (down) に変化	2 (down) のまま	管理者は、インターフェースをダウンすべきことを確認しました。(An administrator has confirmed that the interface should be down.)	クリア
2 (down) から 1 (up) に変化	1 (up) から 2 (down) に変化	インターフェースがダウンしました。(The interface has gone down.)	軽微

表 *1. SNMP* リンク状態ポーリングによって生成されるイベント (続き)

ポーリング間隔の間での **ifAdminStatus MIB** 変数の状況	ポーリング間隔の間での **ifOperStatus MIB** 変数の状況	生成されるイベント	イベント重大度
2 (down) から 1 (up) に変化	2 (down) のまま	管理者はインターフェースをアップするように命令しましたが、アップされていません。(An administrator has instructed the interface to come up, but it has not.)	軽微

関連タスク:

78 ページの『リンク状態ポーリングの構成』

ncp_poller プロセスが、既存のイベントがない場合にリンク状態ポーリングの初期状態を判別する方法を指定します。 **ncp_poller** プロセスは、最初のポーリングを使用してリンク状態ポーリングの初期状態を判別するか、またはクリア状態を想定します。

リモート **ping** ポーリング**:**

リモート ping ポーリング中、エンタープライズ固有のデバイス MIB を使用して、デバイス間の Multi-Protocol Label Switching (MPLS) パスの状況を検査します。特定の MIB モジュールを使用すると、管理ステーションは ping 操作をリモートで開始することができます。リモート SNMP ping 操作では、SNMP を使用して ping 障害をモニターすることができます。

リモート ping ポーリングの操作中、Network Manager IP Edition は、プロバイダー・エッジ (PE) デバイスに、接続先のカスタマー・エッジ (CE) を周期的に ping するように指示します。そのリモート ping 操作の結果、PE デバイスから CE デバイスまでの経路 (MPLS パス) が使用可能であるかダウンしているかについての情報を取得することができます。

制約事項**:** リモート ping 操作は現在、Cisco および Juniper のデバイスのみで使用可能です。

SNMP パスワードの設定については、*IBM Tivoli Network Manager IP Edition* ディスカバリー・ガイドを参照してください。

リモート **ping** ポーリングの前提条件

リモート ping ポーリングを操作できるようにするには、以下の前提条件を満たしていることが必要です。

- PE デバイスへの書き込み権限を保有している必要があります。
- MPLS パスをディスカバーしておく必要があり、またデータを NCIM データベースに転送しておきます。NCIM 内では、データを以下のとおりに配置する必要があります。

  - 仮想プライベート・ネットワーク・ルーター・フォワーディング (VRF) テーブルを VPNRouteForwarding テーブル内にリストする必要があります。
  - PE デバイスから CE デバイスまでのリンクを connects テーブル内にリストする必要があります。
- Juniper リモート ping ポーリングの場合、ビュー・ベースのアクセス制御モデル (VACM) を通して Juniper デバイスへのアクセス権限も必要になります。

VPNRouteForwarding テーブルおよび connects テーブルについての詳細は、*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンスを参照してください。

しきい値ポーリング**:**

しきい値ポーリング中、選択した MIB 変数に定義済み数式が適用され、MIB 変数がしきい値を超えるとイベントが生成されます。MIB 変数の値がしきい値を下回るか、または異なるクリア値を下回ると、クリア・イベントが生成されます。

2 つのしきい値を設定することができます。

生成のしきい値
: 必須: MIB 変数 (複数可) の値がしきい値を超えると、イベントが生成されます。

クリアのしきい値
: オプション: MIB 変数の値がしきい値を下回ると、クリア・イベントが生成されます。

クリアのしきい値を指定しない場合、MIB 変数 (複数可) が生成のしきい値を超えなくなったときは、発行されたイベントが自動的にクリアされます。

しきい値ポーリングの例

モニター管理者は、CPU の使用率が 75% 以上の Cisco 29xx ルーターを識別する必要があります。SNMP ポーリングを使用すると、管理者はネットワーク内のすべての Cisco 29xx ルーターの振る舞いをモニターして、これらの各ルーターの CPU 使用率が 75% を超えるときに当該ルーターに対応するイベントが生成されるように定義することができます。クリアのしきい値を設定して、CPU 使用率が 60% を下回るときに通知が生成されるようにすることもできますが、クリアのしきい値を指定しない場合は CPU 使用率が 75% を超えなくなったときにクリア・イベントが生成されます。

基本しきい値ポーリングと汎用しきい値ポーリング

単純な数式を MIB 変数に適用する場合、またはデバイスおよびインターフェースのレベルでスコープにフィルターをかける場合は、基本しきい値ポーリング を使用します。インターフェース・レベルでフィルターをかけるには、ポーリング定義をインターフェース・フィルター用に設定する必要があります。

複雑な数式の場合、またはスコープにデバイス・レベルのみでフィルターをかける場合は、汎用しきい値ポーリング を使用します。

ポーリング定義タイプ

各ポーリング定義は、1 つのポーリング定義タイプに基づいています。ポーリング定義タイプは、使用するポーリング・メカニズムに従ってグループ化できます。

ポーリング・メカニズムに基づき、ポーリング定義タイプはポーリング操作が使用されるスコープを制限します。

ping ポーリング・メカニズム

ping ポーリング・メカニズムには、以下のポーリング定義タイプがあります。

シャーシ **ping**
: ネットワーク・デバイスの管理インターフェースまたはエンド・ノードのメイン・インターフェースを ping するのに使用されます。

インターフェース **ping**
: デバイス内のインターフェースの ping 操作に使用されます。インターフェース ping ポーリング定義には、オプションでインターフェース・レベル・フィルターがあります。

SNMP ポーリング・メカニズム

SNMP ポーリング・メカニズムには、以下のポーリング定義タイプがあります。

汎用しきい値
: MIB 変数に対して適用する数式を設定するために使用されます。汎用しきい値ポーリング定義は、以下のしきい値から構成されています。

  トリガーしきい値
  : 必須: MIB 変数 (複数可) の値がしきい値を超えると、イベントが生成されます。

  クリアのしきい値
  : オプション: MIB 変数の値がこのしきい値より小さくなると、クリア・イベントが生成されます。

基本しきい値
: 基本しきい値は、単一の MIB 変数または式のポーリング・データを収集するために使用します。収集したデータは、レポートに表示することも、MIB グラフに表示することもできます。ポーリング定義で定義したトリガーしきい値条件が満たされるとイベントが生成され、クリアのしきい値条件が満たされるとイベントがクリアされます。

SNMP リンク状態
: 管理および操作状況をチェックするために使用されます。SNMP リンク状態ポーリング定義には、オプションでインターフェース・レベル・フィルターがあります。

Cisco リモート **ping**
: Cisco 特有の MIB を使用して、デバイスの可用性をチェックするために使用します。

Juniper リモート **ping**
: Juniper 特有の MIB を使用して、デバイスの可用性をチェックします。

データ・ラベル

データ・ラベルは、同じポーリング・データを収集する複数のポーリング定義を単一のレポート内にグループ化できるようにするメカニズムです。データ・ラベルは、基本しきい値ポーリング定義でのみ使用できます。デフォルトで、データ・ラベルはポーリング定義と同じ名前になりますが、この名前はデータ・ラベルの必要に合わせて変更することができます。

以下の例では、データ・ラベルを使用して、単一のレポートで複数のポーリング定義からデータを取得できるようにする方法を示します。多数の Network Manager 要約レポートがデフォルトでデータ・ラベルを使用しています。要約レポートのリストについては、*IBM Tivoli Network Manager IP Edition* 管理ガイドを参照してください。

ベンダー固有の複数のポーリング定義

要約レポートでさまざまなベンダー・デバイスのメモリー使用率 (%) に関するデータを表示するには、ベンダー固有の複数のポーリング定義からポーリング・データを取得する必要があります。それぞれのベンダー固有のポーリング定義内で共通の `memoryPercentageUsage` データ・ラベルを定義することにより、これらの各ポーリング定義で取得されるデータを 1 つのレポート内にグループ化できます。

さまざまなしきい値およびイベント重大度を含むポーリング定義

要約レポートでデバイス・インターフェース上のインバウンド破棄に関するデータを表示するには、複数のポーリング定義からデータを取得します。これらの各ポーリング定義では、同じポーリング・データが収集されますが、このデータには、異なるしきい値とイベント重大度が適用されます。異なる各ポーリング定義内で共通の `ifInDiscards` データ・ラベルを定義することにより、これらの各ポーリング定義で取得されるデータを単一のレポート内にグループ化できます。

レポートとデータ・ラベルのデフォルトのマッピング

以下の表は、要約レポートと、その要約レポートでデフォルトで使用されるデータ・ラベルを示しています。

表 2. レポートとデータ・ラベルのデフォルトのマッピング

レポート	データ・ラベル
ルーターの正常性の要約	memoryPercentageUsage
	cpuBusy
デバイス入り口トラフィックの正常性の要約	snmpInBandwidth
	ifInDiscards
	ifInErrors
デバイス出口トラフィックの正常性の要約	snmpOutBandwidth
	ifOutDiscards
	ifOutErrors
デバイス可用性要約	systemUptime

ping ポーリングのプロパティーおよびメトリック

シャーシおよびインターフェースの ping ポーリングでは、タイムアウト期間や ping 再試行回数などの ping プロパティーを指定できます。応答時間やパケット・ロスなどの ping メトリックを収集することもできます。

シャーシ ping またはインターフェース ping ポーリングを作成するときに、以下の ping プロパティーを指定できます。

タイムアウト
: ポーリング・プロセスが新しい ping パケットを送信する前にターゲット・デバイスからの応答を待つ時間をミリ秒単位で指定します。

再試行
: ポーリング・プロセスを中止する前にターゲット・デバイスの ping を試行する回数を指定します。「パケット・ロス」メトリック収集が有効になっている場合、ポーリング・プロセスは、応答が受信されたかどうかに関係なく、この数の ping パケットを送信します。

ペイロード・サイズ
: ping 要求に使用する ICMP パケットのサイズを選択します。デフォルト (32 バイト) またはカスタム・サイズを選択します。この設定は、`NcPollerSchema.cfg` 構成ファイル内の IcmpData の値をオーバーライドします。

  注意**:**
  32 バイトより小さいサイズを使用すると、パケットがドロップされる場合があります。

シャーシ ping またはインターフェース ping ポーリングを作成するときに、以下の ping メトリックを収集できます。

応答時間
: ping テストの往復時間についてのデータを収集することを選択できます。これはミリ秒単位で測定されます。「パケット・ロス」も収集する場合、これはそれぞれの正常なテストの平均応答時間になります。

パケット・ロス
: ポーリング・プロセスが応答を受信しなかった ping パケットの数についてのデータを収集することを選択できます。これはパーセントとして格納されます。

ポーリング定義内のマルチバイト・データ

中国語 (簡体字) などのマルチバイト文字を使用するドメインで Network Manager を実行している場合は、マルチバイト文字を処理するように Network Manager が構成されていることを確認してから、基本しきい値ポーリング定義または汎用しきい値ポーリング定義を構成する必要があります。

マルチバイト文字を使用するように Network Manager を構成する方法については、*IBM Tivoli Network Manager IP Edition* インストールと構成ガイド を参照してください。

第 2 章 ポーリングの有効化と無効化

Network Manager ポーリングをアクティブにするには、ポーリング・ポリシーを有効にする必要があります。ネットワーク・エンティティーがネットワーク上にない場合は、そのエンティティーをポーリングするポーリング・ポリシーを無効にします。

ヒント: ポーリングを有効にする前にその設定を変更することができます。独自のポーリング・ポリシーを作成する場合は、デフォルトのポーリング・ポリシーを例として使用してください。

デフォルトでは、ディスカバーされたネットワーク・トポロジー内のすべてのデバイス上で有効になっているのはシャーシ ping ポーリングだけです。ただし、デスクトップやプリンターなどのエンド・ノード・デバイスは例外です。

注: 多数のデバイスについてポーリング・ポリシーを有効にする場合のベスト・プラクティスは、ポーリング・ポリシーが完全に有効になるまで待ってから、ネットワーク・ポーリング GUI を使用してポーリング・ポリシーの変更を行うことです。ポーリング・ポリシーを変更すると、ポーリング・エンジン `ncp_poller` が再始動されますが、`ncp_poller` がポーリング・ポリシーを有効化している最中であった場合、これによって予測不能な結果が生じるおそれがあります。ポーリング・ポリシーが有効になっているかどうかを確認するには、ネットワーク・ポーリング GUI の「ポーリング・ポリシーの構成」セクションにある「状況」および「有効」列を使用してください。

ポーリングを有効または無効にするには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 必要なポリシー (複数可) の横にあるチェック・ボックスを選択します。
3. オプション: 選択したポリシーを有効にするには、「選択されたポリシーを有効にする」 をクリックします。
4. オプション: ポリシーを無効にするには、「選択されたポリシーを無効にする 」をクリックします。
5. 「**OK**」をクリックします。

© Copyright IBM Corp. 2006, 2016

第 3 章 ポーリングの作成

既存のデフォルト・ポーリング・ポリシーおよび定義が要件に合わない場合に、ポーリングを作成します。既存のポーリングおよびデフォルト・ポーリングのコピーをカスタマイズするか、新規のポーリングを最初から作成してください。

複数のポーリング定義と完全なスコープ設定機能を含むフル機能のポーリング・ポリシーを作成するには、ポーリング・ポリシー・エディターを使用します。ポーリング・ポリシー・ウィザードを使用して、ガイドに従ってポーリング・ポリシーを作成することもできますが、このウィザードで作成できるのは、単一の既存のポーリング定義と制限されたスコープ設定機能を含む単純なポーリング・ポリシーに限られます。

要確認**:** システムでは、1 つのドメイン内で固有のポーリング・ポリシー名が強制されます。

注**:** 多数のデバイスについてポーリング・ポリシーを有効にする場合のベスト・プラクティスは、ポーリング・ポリシーが完全に有効になるまで待ってから、ネットワーク・ポーリング GUI を使用してポーリング・ポリシーの変更を行うことです。ポーリング・ポリシーを変更すると、ポーリング・エンジン `ncp_poller` が再始動されますが、`ncp_poller` がポーリング・ポリシーを有効化している最中であった場合、これによって予測不能な結果が生じるおそれがあります。ポーリング・ポリシーが有効になっているかどうかを確認するには、ネットワーク・ポーリング GUI の「ポーリング・ポリシーの構成」セクションにある「状況」および「有効」列を使用してください。

関連概念:

10 ページの『ポーリング定義タイプ』
各ポーリング定義は、1 つのポーリング定義タイプに基づいています。ポーリング定義タイプは、使用するポーリング・メカニズムに従ってグループ化できます。

関連タスク:

37 ページの『第 5 章 ポーリングの変更』
ポーリングを変更するには、ポーリング・ポリシーを変更するか、またはポーリングの基礎となるポーリング定義を変更します。

43 ページの『ポーリング定義の変更』
既存のポーリング定義を変更して、ポーリング要件に合わせてカスタマイズします。ポーリング定義の変更は、ポーリング定義エディターで行います。実行するステップは、ポーリング定義タイプ ごとに異なります。

67 ページの『適応ポーリングの作成』
ネットワークのイベントにシステムが動的に反応できるようにするには、適応ポーリングを作成します。

© Copyright IBM Corp. 2006, 2016

フル機能のポーリング・ポリシーの作成

複数のポーリング定義と完全なスコープ設定機能を含むフル機能のポーリング・ポリシーを作成するには、ポーリング・ポリシー・エディターを使用します。

ポーリング・ポリシー・エディターを使用すると、以下の機能を含むポーリング・ポリシーを作成できます。

- 複数のポーリング定義。既存のポーリング定義を使用することも、新規ポーリング定義を作成することもできます。
- ネットワーク・ビュー。ポーリング対象の一連のデバイスを、選択したネットワーク・ビューに含まれるデバイスのみに制限できます。
- デバイス・フィルター。mainNodeDetails テーブルでこの簡易フィルターを使用すると、ネットワーク・ビューで選択されたデバイスのリストを絞り込むことができます。

注: このポーリング・ポリシーに含まれているそれぞれのポーリング定義のスコープをフィルタリングすることによって、ポーリング・ポリシー・スコープをさらに制限できます。ポーリング定義は、デバイス・クラス別およびインターフェース別にフィルタリングできます。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 以下のいずれかを実行して、新規ポリシーを作成します。
   - 新規ポリシーを最初から作成するには、「新規追加」 をクリックします。ポーリング・ポリシー・エディターが開きます。
   - 既存のポリシーを複製するには、以下の手順を実行します。
     a. 「選択」列で、必要な行の横にあるチェック・ボックスを選択して「選択された項目のコピー」 をクリックします。
     b. 「**OK**」をクリックします。このコピーには、*policyname*_1 という規則を使用して名前が付けられます。ここで、*policyname* はコピーされたポリシーの名前です。例えば、ポリシー bgpPeerState をコピーした場合、コピーは bgpPeerState_1 という名前になります。ポーリング・ポリシーはアルファベット順に並べられるため、この例では、コピー bgpPeerState_1 は、リスト内のコピーされたポリシー bgpPeerState の直後に表示されます。
     c. リスト内のポーリング・ポリシーのコピーの名前をクリックして、ポーリング・ポリシー・エディターを開きます。
3. 「ポーリング・ポリシー・プロパティー」タブで、各プロパティーに値を指定します。

   名前　ポーリング・ポリシーに付ける固有の名前を入力します。使用できるのは、英数字、スペース、および下線のみです。

   ポーリング有効
   ポーリング・ポリシーを有効にするには、このチェック・ボックスを選択します。

ポーリング定義

ポーリング・ポリシーの 1 つ以上のポーリング定義を指定するには、このテーブルを使用します。

リフレッシュ 

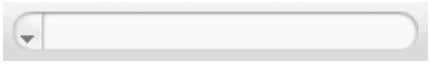
テーブル内のデータをリフレッシュします。これにより、テーブルが更新されて、ログイン以降、または最後に「リフレッシュ」をクリックした時点以降に他のユーザーが行った変更が反映されます。

選択された項目の削除 

選択された行を削除します。

ポーリング定義をこのポリシーに追加します 

「ポーリング定義」パネルを開きます。このパネルでは、ポーリング・ポリシーに追加する 1 つ以上のポーリング定義を指定できます。

「検索」 

「検索」フィールドに入力されているテキストをテーブル内で検索します。デフォルトでは、検索はテーブル内のすべての列に対して実行されます。「検索」フィールドの左側にある下矢印をクリックすることで、検索対象をテーブル内の 1 つ以上の列に制限できます。

- 検索を制限する列に対応するチェック・ボックスを選択します。
- デフォルトの検索設定に戻すには、「すべての列」を選択します。
- 選択が完了したら、「**OK**」をクリックします。

ポーリング定義表

このポーリング・ポリシーに添付されたポーリング定義のリストが、テーブルに示されます。 この表では、以下のアクションを実行できます。設定はこのセッションのみで有効です。

ツールバーを非表示 

ツールバーを非表示にします。ツールバーが非表示になっている場合に、ツールバーを表示するには、「ツールバーを表示」 をクリックします。

ソート列

列を降順でソートするには、列ヘッダーをクリックします。列をもう一度クリックすると、列が昇順でソートされます。以降は、クリックするたびに列の降順と昇順が切り替わります。昇順と降順の意味は、列に含まれるデータのタイプに応じて異なります。

英字データ
昇順では a から z、降順では z から a の順で、データが順序付けされます。

数値データ
昇順では、小さいものから大きいものの順で、データが順序付けされます。降順では、大きいものから小さいものの順で、データが順序付けされます。

アイコン
昇順では、アイコンに関連付けられた値が大きいものから小さいものの順で、アイコンが順序付けされます。降順では、アイコンに関連付けられた値が小さいものから大きいものの順で、アイコンが順序付けされます。各アイコンに関連付けられている値は、下にリストされています。

列のサイズ変更
列を分割している縦線をクリックし、列ヘッダーの右側に向かってドラッグします。

すべて選択/すべてクリア
すべての行を選択するには、このチェック・ボックスを選択します。すべての行が選択されている場合に、すべての行をクリアするには、このチェック・ボックスをクリアします。単一行を選択したり、選択されている単一行をクリアしたりするには、その行の横にあるチェック・ボックスを選択またはクリアします。

保管しますか?
このポーリング定義により収集されたデータをレポート用およびヒストリカル MIB グラフ用に保管するには、このチェック・ボックスを選択します。

注: このオプションは、「基本しきい値」タイプのポーリング定義でのみ使用可能です。

名前　このポーリング・ポリシーに添付されたポーリング定義の名前。このポーリング定義のプロパティーを編集するには、この名前をクリックします。

タイプ
ポーリング定義のタイプ。

状況　ポーリング定義にエラーがあるかどうかを示します。すべての値のリストを、次のテーブルに示します。

表 3. ポーリング定義の状況

状態	値	アイコン	説明
不明	-1		ポーリング定義がまだ実行されていないため、状況が不明です。
エラーなし	0		エラーはありません。 ポーリング定義はエラーなく実行されました。
エラー	0 より大		ポーリング定義にエラーがあります。 ポーリング定義を実行できません。ポーリング定義を使用できるようにするには、その前にエラーを修正する必要があります。エラーを示すポップアップを表示するには、状況アイコンの上にマウスを移動します。

ポーリング間隔
: ポーリングの操作から操作までの間に必要とされる間隔を秒数で指定します。矢印をクリックして値を変更します。

説明
: ポーリング定義の説明。

次のポーリング・プログラム・インスタンスへの割り当て
: 複数のポーリング・プログラム機能の場合のみ: ポーリング・ポリシーを実行するポーリング・プログラムを選択します。定義されているポーリング・プログラムが 1 つしかない場合、このリストは読み取り専用になります。

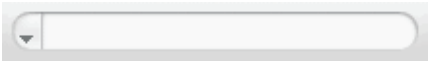
ポリシー・スロットル
: 特定タイプのネットワーク・ビュー、特にイベント・ベース・ネットワーク・ビューのデバイス数は、変動して多くなる場合があります。ポリシーに添付されたネットワーク・ビューのデバイス数の多さが原因で、ポーリング・エンジン ncp_poller が過負荷にならないようにするには、ポーリング・ポリシーに添付されるデバイス数に制限を設ける必要があります。この制限は、ポリシー・スロットルと呼ばれます。

  ポーリングするエンティティーの最大数を指定して、ポーリングを制限します。ポーリング・ポリシーにより、ここで指定された数以下のエンティティーがポーリングされます。

  注: ポリシー・スロットルを無効にするには、この値をゼロに設定します。すべての新規ポーリング・ポリシーは、デフォルトでポリシー・スロットルが無効になっています。

4. ポーリング定義をポーリング・ポリシーに追加することを選択した場合は、「ポーリング定義」パネルで以下のボタンとフィールドを使用して、追加するポーリング定義を指定します。

   リフレッシュ 
   : テーブル内のデータをリフレッシュします。これにより、テーブルが更新されて、ログイン以降、または最後に「リフレッシュ」をクリックした時点以降に他のユーザーが行った変更が反映されます。

「検索」 

「検索」フィールドに入力されているテキストをテーブル内で検索します。デフォルトでは、検索はテーブル内のすべての列に対して実行されます。「検索」フィールドの左側にある下矢印をクリックすることで、検索対象をテーブル内の 1 つ以上の列に制限できます。

- 検索を制限する列に対応するチェック・ボックスを選択します。
- デフォルトの検索設定に戻すには、「すべての列」を選択します。
- 選択が完了したら、「**OK**」をクリックします。

ポーリング定義表

システムで定義されているポーリング定義の完全なリスト。このポーリング・ポリシーに既に添付されているポーリング定義は、そのチェック・ボックスがグレーアウトされています。この表では、以下のアクションを実行できます。設定はこのセッションのみで有効です。

ツールバーを非表示 

ツールバーを非表示にします。ツールバーが非表示になっている場合に、ツールバーを表示するには、「ツールバーを表示」 をクリックします。

ソート列

列を降順でソートするには、列ヘッダーをクリックします。列をもう一度クリックすると、列が昇順でソートされます。以降は、クリックするたびに列の降順と昇順が切り替わります。昇順と降順の意味は、列に含まれるデータのタイプに応じて異なります。

英字データ

昇順では a から z、降順では z から a の順で、データが順序付けされます。

数値データ

昇順では、小さいものから大きいものの順で、データが順序付けされます。降順では、大きいものから小さいものの順で、データが順序付けされます。

アイコン

昇順では、アイコンに関連付けられた値が大きいものから小さいものの順で、アイコンが順序付けされます。降順では、アイコンに関連付けられた値が小さいものから大きいものの順で、アイコンが順序付けされます。各アイコンに関連付けられている値は、下にリストされています。

列のサイズ変更

列を分割している縦線をクリックし、列ヘッダーの右側に向かってドラッグします。

すべて選択/すべてクリア

すべての行を選択するには、このチェック・ボックスを選択します。すべての行が選択されている場合に、すべての行をクリアするには、

このチェック・ボックスをクリアします。単一行を選択したり、選択されている単一行をクリアしたりするには、その行の横にあるチェック・ボックスを選択またはクリアします。

名前　このポーリング・ポリシーに添付されたポーリング定義の名前。このポーリング定義のプロパティーを編集するには、この名前をクリックします。

タイプ
ポーリング定義のタイプ。

説明　ポーリング定義の説明。

ポーリング・データの保管
ポーリング・データを保管し、その後取り出してレポート作成で使用できるようにするには、このチェック・ボックスを選択します。データは、ncpolldata データベースに保管されます。

制約事項: Cisco リモート ping、Juniper リモート ping、および汎用しきい値の各ポーリング定義では、ポーリング・データの保管はサポートされません。

間隔　ポーリングの操作から操作までの間に必要とされる間隔を秒数で指定します。矢印をクリックして値を変更します。

5. 「ネットワーク・ビュー」タブをクリックして、ポーリング・スコープを設定します。「ネットワーク・ビュー」ツリーで、必要なネットワーク・ビューのチェック・ボックスを選択します。 「ネットワーク・ビュー」ツリーには、このポーリング・ポリシーが定義されているネットワーク・ドメインに属するネットワーク・ビューのみが表示されます。ポーリング・ポリシーは 1 つのドメインのみに適用されるため、クロスドメイン・ネットワーク・ビューは表示されません。

   重要: 「すべてのデバイス」オプションを選択すると、「デバイス・フィルター」タブで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。スコープが設定されていない場合、「すべてのデバイス」オプションを選択すると、作成したポーリングで、現行のネットワーク・ドメイン内のすべてのデバイスがポーリングされます。

   このポーリング・ポリシーに含まれているそれぞれのポーリング定義のスコープをフィルタリングすることによって、ポーリング・ポリシー・スコープをさらにフィルタリングできます。ポーリング定義は、デバイス・クラス別およびインターフェース別にフィルタリングできます。

6. オプション: 「デバイス・フィルター」タブをクリックします。これにより、mainNodeDetails デバイス・テーブル上のデバイスのみがフィルタリングされます。以下のいずれかの方法を使用してフィルターを定義します。
   - 「フィルター」列のフィールドに SQL WHERE ステートメントを入力します。

     注: データベースに応じて SQL 構文が異なります。正しい SQL 構文については、使用しているトポロジー・データベースの資料を参照してください。

- 「編集」 をクリックし、フィルター・ビルダーを使用してフィルターをセットアップします。

7. オプション: 「フィルター・ビルダー」の以下の 2 つのタブのいずれかで、必要な照会を作成し、「**OK**」をクリックします。
   - 「基本」タブでは、フィールドを選択し、条件を選択して、値を入力します。ワイルドカードに % 文字を使用します。このフィールドは、選択した属性テーブルに制限されています。
   - 「拡張」タブでは、必要な SQL WHERE ステートメントを入力します。

     注**:** データベースに応じて SQL 構文が異なります。正しい SQL 構文については、使用しているトポロジー・データベースの資料を参照してください。

   「基本」タブに入力する情報は、「拡張」タブに自動的に書き込まれます。
8. オプション: その他の属性テーブルに対するフィルターを追加するには、「新規行の追加」 をクリックし、行を編集してフィルターを作成するためにステップを繰り返します。
9. オプション: 複数のフィルターを結合するには、「すべて」または「いずれか」をクリックします。
   - 「すべて」: 指定したすべてのフィルターと一致するネットワーク・エンティティーのみがポーリングされます。例えば、2 つのフィルターを作成する場合、ネットワーク・エンティティーは両方のフィルターと一致する必要があります。
   - 「いずれか」: 指定したいずれかのフィルターと一致するネットワーク・エンティティーがポーリングされます。
10. 「保存」をクリックします。

関連概念:

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

関連タスク:

75 ページの『ポーリング帯域幅の調整』
ポーリング・エンジン ncp_poller が転送するデータの量と頻度を構成できます。ネットワーク輻輳を回避したり、大量のポーリング・イベントが同時に発生した場合の影響を低減したりするために、ポーリング帯域幅の調整が必要な場合もあります。

関連資料:

221 ページの『付録 D. ポーリング定義の式の構文』
基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成する方法を理解するには、この情報を使用してください。

単純なポーリング・ポリシーの作成

ポーリング・ポリシー・ウィザードを使用すると、ガイドに従ってポーリング・ポリシーを作成できます。ただし、このウィザードを使用して作成できるのは、単一の既存のポーリング定義と制限されたスコープ設定機能を含む単純なポーリング・ポリシーに限られます。

ポーリング・ポリシー・ウィザードを使用すると、以下の制限されたポーリング定義とスコープ設定機能を含む単純なポーリング・ポリシーを作成できます。

- 単一のポーリング定義。既存のポーリング定義を 1 つのみ使用できます。
- ネットワーク・ビュー。ポーリング対象の一連のデバイスを、選択したネットワーク・ビューに含まれるデバイスのみに制限できます。

  制約事項: ポーリング・ポリシー・ウィザードには、ネットワーク・ビューで選択されたデバイスのリストを絞り込むためのデバイス・フィルターはありません。

複数のポーリング定義と完全なスコープ設定機能を含むフル機能のポーリング・ポリシーが必要な場合は、ポーリング・ポリシー・エディターを使用してください。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 「ポーリング構成ウィザードの起動」 をクリックします。
3. 「次へ」をクリックします。「ポーリング・ポリシーの詳細」ページで、以下のように入力します。

   名前　ポーリング・ポリシーの名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   間隔　ポーリングの操作から操作までの間に必要とされる間隔を秒数で指定します。矢印をクリックして値を変更します。

   ポーリング有効
   ポーリングを有効にするかどうかを指定します。ポーリングはデフォルトで有効になっています。ポーリングを無効にするには、このチェック・ボックスをクリアします。

   ポーリング・データの保管
   ポーリング・データを保管し、その後取り出してレポート作成で使用できるようにするには、このチェック・ボックスを選択します。データは、ncpolldata データベースに保管されます。

   制約事項: Cisco リモート ping、Juniper リモート ping、および汎用しきい値の各ポーリング定義では、ポーリング・データの保管はサポートされません。

   定義　リストからポーリング定義を選択します。
4. 「次へ」をクリックします。「ポーリング・ポリシー・スコープの詳細」ページで、必要なネットワーク・ビューのチェック・ボックスを選択します。「ネットワーク・ビュー」ツリーで、必要なネットワーク・ビューのチェック・ボックスを選択します。 「ネットワーク・ビュー」ツリーには、このポーリング・ポリシーが定義されているネットワーク・ドメインに属するネットワーク・ビューの

みが表示されます。ポーリング・ポリシーは 1 つのドメインのみに適用されるため、クロスドメイン・ネットワーク・ビューは表示されません。

重要: 「すべてのデバイス」オプションを選択すると、作成したポーリングで、現行のネットワーク・ドメイン内のすべてのデバイスがポーリングされます。

5. 「次へ」をクリックします。「ポーリング・ポリシーの要約」ページで、指定した情報を確認し、「完了」をクリックします。

関連概念:

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

関連資料:

221 ページの『付録 D. ポーリング定義の式の構文』
基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成する方法を理解するには、この情報を使用してください。

カスタム・データに基づくポーリング・ポリシー作成のためのクイック・リファレンス

ディスカバリーの一環として収集されたカスタム・デバイス・データに基づいてネットワークをポーリングするポーリング・ポリシーを作成することができます。そのためには、ポーリング・ポリシーのスコープが、カスタム・データで編成されたネットワーク・ビューになるように構成します。例えば、デバイスのインターフェースに関連付けられた顧客名や顧客タイプなどのカスタム・データを収集するように、ディスカバリーを構成することができます。その上で、このカスタム・データに基づいてネットワーク・ビューを作成できます。例えば、顧客名で編成したデバイスのネットワーク・ビューを作成することができます。そして、指定した顧客ネットワーク・ビューをポーリングするようにポーリング・ポリシーのスコープを設定することができます。

以下の表に、手順の説明を示します。

表 4. カスタム・データに基づくポーリング・ポリシー作成のためのクイック・リファレンス

アクション	その他の情報
1. カスタム・データでデバイスまたはインターフェースにタグ付けするようにディスカバリーを構成します。 例えば、名前と値のペアのタグ (インターフェースやデバイスに関連付けられた顧客名と顧客住所など) でデバイスまたはインターフェースにタグ付けするようにディスカバリーを構成することができます。これを行うには、右側に示すいずれかの方法を使用します。	ファイル・ファインダーを使用してディスカバリーをシードしている場合は、ファイル・ファインダーが読み取るシード・ファイルに列を追加することによって、名前と値のペアのタグをエンティティーに追加できます。詳しくは、「*IBM Tivoli Network Manager IP Edition* ディスカバリー・ガイド」を参照してください。
	Ping ファインダーを使用してディスカバリーをシードしている場合は、以下のいずれかの方法を使用して名前と値のペアのタグをインターフェースまたはデバイスに割り当てることができます。 • 固有の IP アドレスを使用する。 • フィルター済みの一連の IP アドレスを使用する。を参照してください。 • フィルター方式を選択する場合は、GetCustomTag スティッチャーを使用して、名前と値のペアにおける値の部分を評価するロジックを使用できます。 詳しくは、「*IBM Tivoli Network Manager IP Edition* ディスカバリー・ガイド」を参照してください。
2. NCIM トポロジー・データベースを拡張してカスタム・データを格納します。	例えば、NCIM の entityDetails テーブルに名前と値のペアのタグを追加することができます。詳しくは、「*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンス」を参照してください。
	各デバイスまたはインターフェースで複数のカスタム・データ・フィールドを収集するようにディスカバリーを構成した場合は、新しい NCIM テーブルを作成してそのカスタム・データを格納することができます。詳しくは、「*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンス」を参照してください。

表 4. カスタム・データに基づくポーリング・ポリシー作成のためのクイック・リファレンス (続き)

アクション	その他の情報
3. ネットワーク・ビューを作成するか、ホップ・ビューを参照して、カスタム・データに関連したネットワークの部分を表示します。	フィルター済みネットワーク・ビューや Distinct 動的ネットワーク・ビューを作成して、カスタム・データに関連したネットワークの部分を表示することができます。 • フィルター済みネットワーク・ビューを作成し、カスタム・データを使用するトポロジー・データベース・フィルターに基づいてネットワークの各部分を表示します。例えば、NCIM の entityDetails テーブルを使用するフィルター済みネットワーク・ビューを作成して、指定した顧客名でタグ付けされたインターフェースを含むすべてのデバイスを表示することができます。 • Distinct 動的ネットワーク・ビューを作成して、カスタム・データのカテゴリーおよびサブカテゴリーに基づいてネットワークの各部分を表示します。例えば、NCIM の entityDetails テーブルを使用する Distinct 動的ネットワーク・ビューを作成し、各ビューがカスタム・データのいずれかの顧客名に対応する一連の顧客ビューを生成することができます。各顧客ビューには、その顧客名でタグ付けされたインターフェースを含むすべてのデバイスが表示されます。 詳しくは、「*IBM Tivoli Network Manager IP Edition* ネットワーク可視化セットアップ・ガイド」を参照してください。
	ホップ・ビューで拡張検索を使用して、カスタム・データ内の属性に基づいてデバイスを検索することができます。例えば、ホップ・ビュー内の「エンティティー検索」ウィンドウを使用し、entityDetails テーブルを参照して、特定の場所にいる顧客のすべてのデバイスを取得することができます。詳しくは、「*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイド」を参照してください。
4. 前のステップで作成したネットワーク・ビューをポリシー・スコープとして使用するポーリング・ポリシーを作成します。	ポーリング・ポリシーの作成時には、ポーリング対象のネットワーク・ビューを指定することができます。前のステップで作成したネットワーク・ビューのうち 1 つ以上を選択することにより、指定した顧客名でタグ付けされたインターフェースを含むデバイスをポーリングすることができます。詳しくは、「*IBM Tivoli Network Manager IP Edition* イベント管理ガイド」を参照してください。

第 4 章 新規ポーリング定義の作成

新規ポーリング定義を作成するステップをガイドするポーリング定義エディターを使用します。

ポーリング定義を作成または変更する前に、既存のポーリング定義を表示して、それを新規のポーリング定義を作成するためのテンプレートとして使用できないかどうか判断してください。

要確認**:** システムでは、1 つのドメイン内で固有のポーリング定義名が強制されます。

さまざまなポーリング定義タイプがあるため、ポーリング定義エディターに表示されるページは、選択するポーリング定義タイプごとに異なります。

関連タスク:

43 ページの『ポーリング定義の変更』
既存のポーリング定義を変更して、ポーリング要件に合わせてカスタマイズします。ポーリング定義の変更は、ポーリング定義エディターで行います。実行するステップは、ポーリング定義タイプ ごとに異なります。

15 ページの『第 3 章 ポーリングの作成』
既存のデフォルト・ポーリング・ポリシーおよび定義が要件に合わない場合に、ポーリングを作成します。既存のポーリングおよびデフォルト・ポーリングのコピーをカスタマイズするか、新規のポーリングを最初から作成してください。

関連資料:

211 ページの『付録 B. デフォルトのポーリング定義』
Network Manager IP Editionには、一般的なポーリング要件に対応したさまざまなデフォルトのポーリング定義が用意されています。

基本しきい値ポーリング定義の作成

MIB 変数に対して単純な式を実行したり、インターフェース・レベルのフィルターを使用するしきい値ポーリングを作成するには、基本しきい値ポーリング定義を作成します。

ポーリング定義を作成または変更する前に、既存のポーリング定義を表示して、それを新規のポーリング定義を作成するためのテンプレートとして使用できないかどうか判断してください。

基本しきい値ポーリング定義を作成するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 「新規追加」 をクリックします。 「新規ポーリング定義タイプの選択」ページが表示されます。
3. リストから「**基本しきい値**」を選択して、「**OK**」をクリックします。

© Copyright IBM Corp. 2006, 2016

4. ポーリング定義エディター の「一般」タブで、「一般プロパティー」の各フィールドに以下のように入力します。

   名前　ポーリング定義の固有の名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   タイプ

   このフィールドは使用不可です。このポーリング定義が、有効にされたポーリング・ポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。イベント ID に割り当てられる値は POLL-polldef です。

   イベント **ID**

   このフィールドは使用不可です。このポーリング定義が、有効にされたポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。「イベント **ID**」フィールドには、以下のようにデータが取り込まれます。

   - これが新しいポーリング定義である場合、「イベント **ID**」フィールドには、値 POLL-*polldef* が取り込まれます。ここで、*polldef* は、現在のポーリング定義の名前です。
   - 既存のポーリング定義をコピーすることによってポーリング定義を作成した場合、「イベント **ID**」には、コピーされたポーリング定義と同じ値が入ります。

   注**:** 一部のより古いデフォルトのポーリングの「イベント **ID**」フィールドには、POLL-*polldef* 命名規則が使用されていないものがあります。

   イベント重大度

   重大度として有効な数値を指定します。重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。使用可能な重大度レベルのリストについては、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

   説明　ポーリング定義の簡略説明を入力します。

   データ・ラベル

   データ・ラベル・リストをクリックし、リストからいずれかのデータ・ラベルを選択します。デフォルトで、データ・ラベルは、現在のポーリング定義と同じ名前になります。新規データ・ラベルを定義するには、「<新規データ・ラベルの追加>」を選択します。リストの右側のフィールドがアクティブになります。このフィールドに、新規データ・ラベルの名前を入力します。

5. 「クラス」タブをクリックします。「クラス」ツリーで、必要なクラスのチェック・ボックスを選択します。

   重要**:** すべてのクラスのチェック・マークを外すと、このポーリング定義を使用するポーリング・ポリシーで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。

6. オプション: 「インターフェース・フィルター」タブをクリックして、必須フィールドのフィルターを作成します。 「テーブル」フィールドには、interfaces テーブルのデータが事前に取り込まれています。
7. 「ポーリング・データ」タブをクリックして、必要な式を指定します。
   - MIB オブジェクト ID (OID) を指定するには、「単一 **OID**」を選択します。必須 MIB 変数の現行値またはデルタ値を指定して、次のフィールドに変数を入力します。
   - 複合式を指定するには、「式」を選択して、フィールドに式を入力します。

   MIB ツリーから直接変数を選択するには、「**MIB** オブジェクトの追加」 をクリックします。MIB ツリーから、選択した MIB 変数の現行値または前の値を指定したり、変数の現行値を SNMP 索引に解決したりすることができます。
8. 「しきい値」タブをクリックして、イベントのトリガーおよびイベントのクリア用の式を指定します。 「ポーリング・データ」タブで指定した MIB OID または式は、自動的に式に書き込まれます。
   a. 「トリガーしきい値」領域でリストから条件を選択し、MIB OID をフィルターする値を入力します。
   b. 「説明」フィールドに、トリガー式の分かりやすい説明を入力します。括弧書きの MIB 変数を説明に追加します。 イベントが発生すると、説明が AEL に表示されます。 例えば、次のようになります。

      ```
      CPU usage high (avgBusy5=)
      ```
   c. 基礎 EVAL ステートメントを説明に挿入するには、右括弧の内側にカーソルを置いて「**MIB** オブジェクトの追加  」をクリックし、指定した変数にナビゲートします。変数の現行値または前の値を評価するのか、値を SNMP 索引に解決するのかを指定して、「**OK**」をクリックします。 ステートメントが挿入されます。例:

      ```
      CPU usage high (avgBusy5=eval(text,"&SNMP.VALUE.sysName"))
      ```
   d. 「クリアしきい値」領域について、ステップ 8a から 8c を繰り返します。
9. 「保存」をクリックします。

関連概念:

12 ページの『ポーリング定義内のマルチバイト・データ』
中国語 (簡体字) などのマルチバイト文字を使用するドメインで Network Manager を実行している場合は、マルチバイト文字を処理するように Network Manager が構成されていることを確認してから、基本しきい値ポーリング定義または汎用しきい値ポーリング定義を構成する必要があります。

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

関連資料:

54 ページの『基本しきい値の式の例』
この基本しきい値の式の例を使用して、複雑な基本しきい値の式を構成する方法を理解してください。

221 ページの『付録 D. ポーリング定義の式の構文』
基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑

なしきい値の式を作成する方法を理解するには、この情報を使用してください。

汎用しきい値ポーリング定義の作成

ポーリング定義エディターを使用して、新規の汎用しきい値ポーリング定義を作成します。

汎用しきい値定義を作成する場合は、式を設定して複数の式を結合します。

汎用しきい値ポーリング定義を作成するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 「新規追加」  をクリックします。 「新規ポーリング定義タイプの選択」ページが表示されます。
3. リストから「**汎用しきい値**」を選択して、「**OK**」をクリックします。
4. ポーリング定義エディター の「一般」タブで、「一般プロパティー」の各フィールドに以下のように入力します。

   名前　ポーリング定義の固有の名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   タイプ

   このフィールドは使用不可です。このポーリング定義が、有効にされたポーリング・ポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。イベント ID に割り当てられる値は POLL-polldef です。

   イベント **ID**

   このフィールドは使用不可です。このポーリング定義が、有効にされたポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。「イベント **ID**」フィールドには、以下のようにデータが取り込まれます。

   - これが新しいポーリング定義である場合、「イベント **ID**」フィールドには、値 `POLL-polldef` が取り込まれます。ここで、*polldef* は、現在のポーリング定義の名前です。
   - 既存のポーリング定義をコピーすることによってポーリング定義を作成した場合、「イベント **ID**」には、コピーされたポーリング定義と同じ値が入ります。

   注**:** 一部のより古いデフォルトのポーリングの「イベント **ID**」フィールドには、`POLL-polldef` 命名規則が使用されていないものがあります。

   イベント重大度

   重大度として有効な数値を指定します。重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。使用可能な重大度レベルのリストについては、*IBM*

Tivoli Network Manager IP Edition ネットワーク・トラブル・シューティング・ガイドを参照してください。

説明　ポーリング定義の簡略説明を入力します。

5. 「クラス」タブをクリックします。「クラス」ツリーで、必要なクラスのチェック・ボックスを選択します。

   重要: すべてのクラスのチェック・マークを外すと、このポーリング定義を使用するポーリング・ポリシーで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。

6. オプション: 「インターフェース・フィルター」タブをクリックして、必須フィールドのフィルターを作成します。 「テーブル」フィールドには、`interfaces` テーブルのデータが事前に取り込まれています。
7. 「トリガーしきい値」タブをクリックします。以下のいずれかの方式を使用して、しきい値を指定する式を作成します。
   - 「基本」領域で、フィールドとオプションを使用して式を作成します。MIB ツリーから値を選択するには、「**MIB** ツリーを開く」 をクリックします。
   - 「拡張」領域で、オブジェクト照会言語 (OQL) で必要な **eval** ステートメントを入力します。
8. 生成されるイベントの AEL に表示されるメッセージを指定します。
   a. 「イベントの記述」フィールドに、メッセージを入力します。
   b. フィールドに MIB 変数を挿入するには、「**MIB** ツリーを開く」 をクリックします。現行の SNMP 値または前の SNMP 値か、SNMP 索引のいずれかを組み込むようにメッセージを設定して、「**OK**をクリックします。
9. 必須: 「しきい値のクリア」をクリックします。すべての汎用しきい値ポーリング定義には、クリアのしきい値が必要です。以下のいずれかの方式を使用して、しきい値を指定する式を作成します。
   - 「基本」領域で、フィールドとオプションを使用して式を作成します。MIB ツリーから値を選択するには、「**MIB** ツリーを開く」 をクリックします。
   - 「拡張」領域で、オブジェクト照会言語 (OQL) で必要な **eval** ステートメントを入力します。

   ヒント: しきい値が手動でクリアされるようにしたい場合は、到達されることのないクリアのしきい値を作成してください。
10. 生成されるイベントの AEL に表示されるメッセージを指定します。
    a. 「イベントの記述」フィールドに、メッセージを入力します。
    b. フィールドに MIB 変数を挿入するには、「**MIB** ツリーを開く」 をクリックします。現行の SNMP 値または前の SNMP 値か、SNMP 索引のいずれかを組み込むようにメッセージを設定して、「**OK**をクリックします。
11. 「保存」をクリックし、次に「**OK**」をクリックします。

ポーリング定義がリストの下部に追加されます。

関連概念:

12 ページの『ポーリング定義内のマルチバイト・データ』
中国語 (簡体字) などのマルチバイト文字を使用するドメインで Network Manager を実行している場合は、マルチバイト文字を処理するように Network Manager が構成されていることを確認してから、基本しきい値ポーリング定義または汎用しきい値ポーリング定義を構成する必要があります。

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

関連資料:

54 ページの『汎用しきい値の式の例』
この汎用しきい値の式の例を使用して、複雑な汎用しきい値の式を構成する方法を理解してください。

221 ページの『付録 D. ポーリング定義の式の構文』
基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成する方法を理解するには、この情報を使用してください。

シャーシ ping およびインターフェース ping ポーリング定義の作成

ポーリング定義エディターを使用して、シャーシ ping およびインターフェース ping ポーリング定義タイプを作成します。

ポーリング定義を作成するステップは、上記のすべてのポーリング定義タイプで同じです。

シャーシ ping またはインターフェース ping ポーリング定義を作成するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 「新規追加」 をクリックします。 「新規ポーリング定義タイプの選択」ページが表示されます。
3. リストからシャーシ ping およびインターフェース ping を選択します。
4. ポーリング定義エディター の「一般」タブで、「一般プロパティー」の各フィールドに以下のように入力します。

   名前　ポーリング定義の固有の名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   タイプ
   このフィールドは使用不可です。このポーリング定義が、有効にされたポーリング・ポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。イベント ID に割り当てられる値は POLL-polldef です。

   イベント **ID**
   このフィールドは使用不可です。このポーリング定義が、有効にされたポリシーの一部として組み込まれると、ポーリング・エンジン

(ncp_poller) は、このフィールドに自動的にデータを取り込みます。「イベント **ID**」フィールドには、以下のようにデータが取り込まれます。

- これが新しいポーリング定義である場合、「イベント **ID**」フィールドには、値 POLL-*polldef* が取り込まれます。ここで、*polldef* は、現在のポーリング定義の名前です。
- 既存のポーリング定義をコピーすることによってポーリング定義を作成した場合、「イベント **ID**」には、コピーされたポーリング定義と同じ値が入ります。

注**:** 一部のより古いデフォルトのポーリングの「イベント **ID**」フィールドには、POLL-*polldef* 命名規則が使用されていないものがあります。

イベント重大度
: 重大度として有効な数値を指定します。重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。使用可能な重大度レベルのリストについては、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

説明　ポーリング定義の簡略説明を入力します。

5. 「クラス」タブをクリックします。「クラス」ツリーで、必要なクラスのチェック・ボックスを選択します。

   重要**:** すべてのクラスのチェック・マークを外すと、このポーリング定義を使用するポーリング・ポリシーで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。

6. オプション: 「インターフェース・フィルター」タブをクリックして、必須フィールドのフィルターを作成します。 「テーブル」フィールドには、interfaces テーブルのデータが事前に取り込まれています。
7. 「**ping**」タブをクリックして、「**ping** プロパティー」フィールドに以下のように入力します。

   タイムアウト
   : ポーリング・プロセスが新しい ping パケットを送信する前にターゲット・デバイスからの応答を待つ時間をミリ秒単位で指定します。

   再試行
   : ポーリング・プロセスを中止する前にターゲット・デバイスの ping を試行する回数を指定します。「パケット・ロス」メトリック収集が有効になっている場合、ポーリング・プロセスは、応答が受信されたかどうかに関係なく、この数の ping パケットを送信します。

   ping メトリックの収集

   応答時間
   : デバイスが ping 要求に応答するために要した時間を収集するには、このボックスにチェック・マークを付けます。

   パケット・ロス
   : 失われたパケットに関するデータを収集するには、このボックスにチェック・マークを付けます。

ペイロード・サイズ

ping 要求に使用する ICMP パケットのサイズを選択します。デフォルト (32 バイト) またはカスタム・サイズを選択します。この設定は、NcPollerSchema.cfg 構成ファイル内の IcmpData の値をオーバーライドします。

注意:
32 バイトより小さいサイズを使用すると、パケットがドロップされる場合があります。

8. 「保存」をクリックし、次に「**OK**」をクリックします。

関連概念:

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

リモート ping ポーリング定義およびリンク状態ポーリング定義の作成

ポーリング定義エディターを使用して、新規ポーリング定義を作成します。指定できるポーリング定義タイプは、Cisco リモート ping、Juniper リモート ping、SNMP リンク状態です。

ポーリング定義を作成するステップは、上記のすべてのポーリング定義タイプで同じです。

リモート ping ポーリング定義または SNMP リンク状態ポーリング定義を作成するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 「新規追加」  をクリックします。 「新規ポーリング定義タイプの選択」ページが表示されます。
3. リストから必要な定義タイプを選択します。
   - Cisco リモート ping
   - Juniper リモート ping
   - SNMP リンク状態
4. 「**OK**」をクリックします。
5. ポーリング定義エディター の「一般」タブで、「一般プロパティー」の各フィールドに以下のように入力します。

   名前　ポーリング定義の固有の名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   タイプ

   このフィールドは使用不可です。このポーリング定義が、有効にされたポーリング・ポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。イベント ID に割り当てられる値は POLL-polldef です。

イベント **ID**

このフィールドは使用不可です。このポーリング定義が、有効にされたポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。「イベント **ID**」フィールドには、以下のようにデータが取り込まれます。

- これが新しいポーリング定義である場合、「イベント **ID**」フィールドには、値 POLL-*polldef* が取り込まれます。ここで、*polldef* は、現在のポーリング定義の名前です。
- 既存のポーリング定義をコピーすることによってポーリング定義を作成した場合、「イベント **ID**」には、コピーされたポーリング定義と同じ値が入ります。

注**:** 一部のより古いデフォルトのポーリングの「イベント **ID**」フィールドには、POLL-*polldef* 命名規則が使用されていないものがあります。

イベント重大度

重大度として有効な数値を指定します。重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。使用可能な重大度レベルのリストについては、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

説明　ポーリング定義の簡略説明を入力します。

6. 「クラス」タブをクリックします。「クラス」ツリーで、必要なクラスのチェック・ボックスを選択します。

   重要**:** すべてのクラスのチェック・マークを外すと、このポーリング定義を使用するポーリング・ポリシーで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。

7. オプション: 「インターフェース・フィルター」タブをクリックして、必須フィールドのフィルターを作成します。 「テーブル」フィールドには、interfaces テーブルのデータが事前に取り込まれています。
8. 「保存」をクリックし、次に「**OK**」をクリックします。

関連概念:

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

第 5 章 ポーリングの変更

ポーリングを変更するには、ポーリング・ポリシーを変更するか、またはポーリングの基礎となるポーリング定義を変更します。

関連概念:

2 ページの『ポーリング・ポリシー』
ポーリング・ポリシーは、ネットワーク・ポーリング操作のプロパティーすべてを含んでいます。ポーリング・ポリシーでは、デバイスのポーリング頻度、ポーリング実行のために採用するポーリング・メカニズムのタイプ、およびポーリングするデバイスを指定します。

4 ページの『ポーリング定義』
ポーリング定義は、ネットワーク・エンティティーをどのようにポーリングするかを決定します。各ポーリング・ポリシーを、少なくとも 1 つのポーリング定義に関連付ける必要があります。1 つのポーリング・ポリシーを複数のポーリング定義に関連付けることができます。

関連タスク:

15 ページの『第 3 章 ポーリングの作成』
既存のデフォルト・ポーリング・ポリシーおよび定義が要件に合わない場合に、ポーリングを作成します。既存のポーリングおよびデフォルト・ポーリングのコピーをカスタマイズするか、新規のポーリングを最初から作成してください。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』
Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

ポーリング・ポリシーの変更

ポーリング・ポリシー・エディターを使用して、既存のポーリング・ポリシーの設定を変更します。

ポーリング・ポリシーを作成または変更する前に、既存のポーリング・ポリシーを表示して、そのポーリングを新規のポーリング・ポリシーを作成するためのテンプレートとして使用できないかどうか判断してください。

注**:** 多数のデバイスについてポーリング・ポリシーを有効にする場合のベスト・プラクティスは、ポーリング・ポリシーが完全に有効になるまで待ってから、ネットワーク・ポーリング GUI を使用してポーリング・ポリシーの変更を行うことです。ポーリング・ポリシーを変更すると、ポーリング・エンジン `ncp_poller` が再始動されますが、`ncp_poller` がポーリング・ポリシーを有効化している最中であった場合、これによって予測不能な結果が生じるおそれがあります。ポーリング・ポリシーが有効になっているかどうかを確認するには、ネットワーク・ポーリング GUI の「ポーリング・ポリシーの構成」セクションにある「状況」および「有効」列を使用してください。

ポーリング・ポリシーを変更するには、以下のようにします。

© Copyright IBM Corp. 2006, 2016

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 必要なポーリング・ポリシーをクリックします。 ポーリング・ポリシー・エディターが表示、選択したポーリング・ポリシーの設定が自動的にフィールドにロードされます。
3. 「ポーリング・ポリシー・プロパティー」で、以下のフィールドに値を指定します。

   名前
   ポーリング・ポリシーに付ける固有の名前を入力します。使用できるのは、英数字、スペース、および下線のみです。

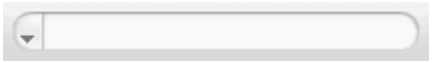
   ポーリング有効
   ポーリング・ポリシーを有効にするには、このチェック・ボックスを選択します。

   ポーリング定義
   ポーリング・ポリシーの 1 つ以上のポーリング定義を指定するには、このテーブルを使用します。

   リフレッシュ 
   テーブル内のデータをリフレッシュします。これにより、テーブルが更新されて、ログイン以降、または最後に「リフレッシュ」をクリックした時点以降に他のユーザーが行った変更が反映されます。

   選択された項目の削除 
   選択された行を削除します。

   ポーリング定義をこのポリシーに追加します 
   「ポーリング定義」パネルを開きます。このパネルでは、ポーリング・ポリシーに追加する 1 つ以上のポーリング定義を指定できます。

   「検索」 
   「検索」フィールドに入力されているテキストをテーブル内で検索します。デフォルトでは、検索はテーブル内のすべての列に対して実行されます。「検索」フィールドの左側にある下矢印をクリックすることで、検索対象をテーブル内の 1 つ以上の列に制限できます。
   - 検索を制限する列に対応するチェック・ボックスを選択します。
   - デフォルトの検索設定に戻すには、「すべての列」を選択します。
   - 選択が完了したら、「**OK**」をクリックします。

   ポーリング定義表
   このポーリング・ポリシーに添付されたポーリング定義のリストが、テーブルに示されます。 この表では、以下のアクションを実行できます。設定はこのセッションのみで有効です。

ツールバーを非表示 

ツールバーを非表示にします。ツールバーが非表示になっている場合に、ツールバーを表示するには、「ツールバーを表示」 をクリックします。

ソート列

列を降順でソートするには、列ヘッダーをクリックします。列をもう一度クリックすると、列が昇順でソートされます。以降は、クリックするたびに列の降順と昇順が切り替わります。昇順と降順の意味は、列に含まれるデータのタイプに応じて異なります。

英字データ

昇順では a から z、降順では z から a の順で、データが順序付けされます。

数値データ

昇順では、小さいものから大きいものの順で、データが順序付けされます。降順では、大きいものから小さいものの順で、データが順序付けされます。

アイコン

昇順では、アイコンに関連付けられた値が大きいものから小さいものの順で、アイコンが順序付けされます。降順では、アイコンに関連付けられた値が小さいものから大きいものの順で、アイコンが順序付けされます。各アイコンに関連付けられている値は、下にリストされています。

列のサイズ変更

列を分割している縦線をクリックし、列ヘッダーの右側に向かってドラッグします。

すべて選択/すべてクリア

すべての行を選択するには、このチェック・ボックスを選択します。すべての行が選択されている場合に、すべての行をクリアするには、このチェック・ボックスをクリアします。単一行を選択したり、選択されている単一行をクリアしたりするには、その行の横にあるチェック・ボックスを選択またはクリアします。

保管しますか**?**

このポーリング定義により収集されたデータをレポート用およびヒストリカル MIB グラフ用に保管するには、このチェック・ボックスを選択します。

注**:** このオプションは、「基本しきい値」タイプのポーリング定義でのみ使用可能です。

名前　このポーリング・ポリシーに添付されたポーリング定義の名前。このポーリング定義のプロパティーを編集するには、この名前をクリックします。

タイプ
ポーリング定義のタイプ。

状況　ポーリング定義にエラーがあるかどうかを示します。すべての値のリストを、次のテーブルに示します。

表 5. ポーリング定義の状況

状態	値	アイコン	説明
不明	-1		ポーリング定義がまだ実行されていないため、状況が不明です。
エラーなし	0		エラーはありません。 ポーリング定義はエラーなく実行されました。
エラー	0 より大		ポーリング定義にエラーがあります。 ポーリング定義を実行できません。ポーリング定義を使用できるようにするには、その前にエラーを修正する必要があります。エラーを示すポップアップを表示するには、状況アイコンの上にマウスを移動します。

ポーリング間隔
ポーリングの操作から操作までの間に必要とされる間隔を秒数で指定します。矢印をクリックして値を変更します。

説明　ポーリング定義の説明。

次のポーリング・プログラム・インスタンスへの割り当て
複数のポーリング・プログラム機能の場合のみ: ポーリング・ポリシーを実行するポーリング・プログラムを選択します。定義されているポーリング・プログラムが 1 つしかない場合、このリストは読み取り専用になります。

ポリシー・スロットル
特定タイプのネットワーク・ビュー、特にイベント・ベース・ネットワーク・ビューのデバイス数は、変動して多くなる場合があります。ポリシーに添付されたネットワーク・ビューのデバイス数の多さが原因で、ポーリング・エンジン ncp_poller が過負荷にならないようにするには、ポーリング・ポリシーに添付されるデバイス数に制限を設ける必要があります。この制限は、ポリシー・スロットルと呼ばれます。

ポーリングするエンティティーの最大数を指定して、ポーリングを制限します。ポーリング・ポリシーにより、ここで指定された数以下のエンティティーがポーリングされます。

注: ポリシー・スロットルを無効にするには、この値をゼロに設定します。すべての新規ポーリング・ポリシーは、デフォルトでポリシー・スロットルが無効になっています。

4. 「ネットワーク・ビュー」タブをクリックします。「ネットワーク・ビュー」ツリーで、必要なネットワーク・ビューのチェック・ボックスを選択します。

「ネットワーク・ビュー」ツリーには、このポーリング・ポリシーが定義されているネットワーク・ドメインに属するネットワーク・ビューのみが表示されます。

重要: 「すべてのデバイス」オプションを選択すると、「デバイス・フィルター」タブで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。スコープが設定されていない場合、「すべてのデバイス」オプションを選択すると、作成したポーリングで、現行のネットワーク・ドメイン内のすべてのデバイスがポーリングされます。

5. オプション: 「デバイス・フィルター」タブをクリックします。これにより、mainNodeDetails デバイス・テーブル上のデバイスのみがフィルタリングされます。以下のいずれかの方法を使用してフィルターを定義します。
   - 「フィルター」列のフィールドに SQL WHERE ステートメントを入力します。

     注: データベースに応じて SQL 構文が異なります。正しい SQL 構文については、使用しているトポロジー・データベースの資料を参照してください。

   - 「編集」  をクリックし、フィルター・ビルダーを使用してフィルターをセットアップします。
6. 「フィルター・ビルダー」の以下の 2 つのタブのいずれかで、必要な照会を作成し、「**OK**」をクリックします。
   - 「基本」タブでは、フィールドを選択し、条件を選択して、値を入力します。ワイルドカードに % 文字を使用します。このフィールドは、選択した属性テーブルに制限されています。
   - 「拡張」タブでは、必要な SQL WHERE ステートメントを入力します。

     注: データベースに応じて SQL 構文が異なります。正しい SQL 構文については、使用しているトポロジー・データベースの資料を参照してください。

   「基本」タブに入力する情報は、「拡張」タブに自動的に書き込まれます。
7. その他の属性テーブルに対するフィルターを追加するには、「新規行の追加」  をクリックし、行を編集してフィルターを作成するためにステップを繰り返します。
8. 複数のフィルターを結合するには、「すべて」または「いずれか」をクリックします。
   - 「すべて」: 指定したすべてのフィルターと一致するネットワーク・エンティティーのみがポーリングされます。例えば、2 つのフィルターを作成する場合、ネットワーク・エンティティーは両方のフィルターと一致する必要があります。
   - 「いずれか」: 指定したいずれかのフィルターと一致するネットワーク・エンティティーがポーリングされます。
9. 「保存」をクリックします。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』
Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

ポーリング・ポリシーの例

このポーリング・ポリシーのカスタマイズ例を使用すると、既存のポリシーをコピーして、クラス C のサブネット内の特定のデバイスをポーリングするようにカスタマイズできるようになります。

シナリオ

以下の要件を満たすようにポーリング・ポリシーをカスタマイズする必要があります。

- そのポーリング・ポリシーでは、ネットワークをチェックして、IP アドレス 9.1.2.* または 10.123.46* のクラス C サブネットに存在するデバイスを確認する必要があります。
- そのポーリング・ポリシーでは、60 秒の間隔でネットワークをチェックする必要があります。
- そのポーリング・ポリシーでは、ポリシーの保存後すぐにネットワークに対するポーリングを開始する必要があります。

必要な設定

前のシナリオで説明した要件を満たすポーリングを作成するには、以下の設定を行います。

1. 「ポーリング・ポリシーの構成」ページで、ciscoMemoryPctgUsage ポーリング・ポリシーのコピーを作成します。ポーリング・ポリシーのコピーがポーリング・ポリシー・テーブルの新規行として表示されます。
2. ciscoMemoryPctgUsage ポーリング・ポリシーのコピーを含む行を見つけ、このポーリング・ポリシーのコピーの名前をクリックします。これにより、ポーリング・ポリシー・エディターでポーリング・ポリシーのコピーを編集できるようになります。
3. 「ポーリング・ポリシー・プロパティー」タブを以下のように設定します。

   名前　意味のある名前 (`ciscoMemoryPctgUsage for class C subnets with 9.1.2* and 10.123.46*` など) を入力します。

   ポーリング有効:
   このチェック・ボックスを選択します。

   ポーリング間隔
   「ポーリング定義」テーブルでスクロールし、「ポーリング間隔」列に `60` と入力します。
4. 「ネットワーク・ビュー」タブで「すべてのデバイス」が選択されていることを確認します。
5. 「デバイス・フィルター」タブを以下のように設定します。
   a. 「いずれか」を選択します。
   b. mainNodeDetails テーブルの各フィールドを対象にしたフィルターを指定するために、「フィルター・ビルダーを開く」 をクリックします。
   c. 「フィルター・ビルダー」の「基本」タブの各フィールドを以下のように入力します。

```
「フィールド」        「条件」       「値」
ipAddress   like          9.1.2.%
```

d. 「**OK**」をクリックします。

e. 「追加  」をクリックします。

f. mainNodeDetails テーブルの各フィールドを対象にした別のフィルターを指定するために、「フィルター・ビルダーを開く」  をクリックします。

g. 「フィルター・ビルダー」の「基本」タブの各フィールドを以下のように入力します。

```
「フィールド」        「条件」       「値」
ipAddress   like          10.123.46.%
```

h. 「**OK**」をクリックします。

6. 「保存」をクリックします。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』
Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

ポーリング定義の変更

既存のポーリング定義を変更して、ポーリング要件に合わせてカスタマイズします。ポーリング定義の変更は、ポーリング定義エディターで行います。実行するステップは、ポーリング定義タイプ ごとに異なります。

ポーリング定義を作成または変更する前に、既存のポーリング定義を表示して、それを新規のポーリング定義を作成するためのテンプレートとして使用できないかどうか判断してください。

関連タスク:

27 ページの『第 4 章 新規ポーリング定義の作成』
新規ポーリング定義を作成するステップをガイドするポーリング定義エディターを使用します。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』
Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

基本しきい値ポーリング定義の変更

ポーリング定義エディターを使用して、基本しきい値ポーリング定義を変更します。

ポーリング定義の一部の一般プロパティー、およびポーリング定義タイプに関連付けられているプロパティーを変更できます。ただし、ポーリング定義タイプを変更することはできません。

基本しきい値ポーリング定義を変更するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 必要なポーリング定義をクリックします。 ポーリング定義は、基本しきい値というポーリング定義タイプになっている必要があります。
3. ポーリング定義エディター の「一般」タブで、「一般プロパティー」の各フィールドに以下のように入力します。

   名前　ポーリング定義の固有の名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   タイプ

   このフィールドは使用不可です。このポーリング定義が、有効にされたポーリング・ポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。イベント ID に割り当てられる値は POLL-polldef です。

   イベント **ID**

   このフィールドは使用不可です。このポーリング定義が、有効にされたポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。「イベント **ID**」フィールドには、以下のようにデータが取り込まれます。

   - これが新しいポーリング定義である場合、「イベント **ID**」フィールドには、値 POLL-*polldef* が取り込まれます。ここで、*polldef* は、現在のポーリング定義の名前です。
   - 既存のポーリング定義をコピーすることによってポーリング定義を作成した場合、「イベント **ID**」には、コピーされたポーリング定義と同じ値が入ります。

   注**:** 一部のより古いデフォルトのポーリングの「イベント **ID**」フィールドには、POLL-*polldef* 命名規則が使用されていないものがあります。

   イベント重大度

   重大度として有効な数値を指定します。重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。使用可能な重大度レベルのリストについては、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

   説明　ポーリング定義の簡略説明を入力します。

データ・ラベル

データ・ラベル・リストをクリックし、リストからいずれかのデータ・ラベルを選択します。デフォルトで、データ・ラベルは、現在のポーリング定義と同じ名前になります。新規データ・ラベルを定義するには、「**<新規データ・ラベルの追加>**」を選択します。リストの右側のフィールドがアクティブになります。このフィールドに、新規データ・ラベルの名前を入力します。

4. 「クラス」タブをクリックします。「クラス」ツリーで、必要なクラスのチェック・ボックスを選択します。

   重要**:** すべてのクラスのチェック・マークを外すと、このポーリング定義を使用するポーリング・ポリシーで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。

5. オプション: 「インターフェース・フィルター」タブをクリックして、必須フィールドのフィルターを作成します。 「テーブル」フィールドには、`interfaces` テーブルのデータが事前に取り込まれています。

6. 「ポーリング・データ」タブをクリックして、必要な式を指定します。

   - MIB オブジェクト ID (OID) を指定するには、「単一 **OID**」を選択します。必須 MIB 変数の現行値またはデルタ値を指定して、次のフィールドに変数を入力します。
   - 複合式を指定するには、「式」を選択して、フィールドに式を入力します。

   MIB ツリーから直接変数を選択するには、「**MIB** オブジェクトの追加」 をクリックします。MIB ツリーから、選択した MIB 変数の現行値または前の値を指定したり、変数の現行値を SNMP 索引に解決したりすることができます。

7. 「しきい値」タブをクリックして、イベントのトリガーおよびイベントのクリア用の式を指定します。 「ポーリング・データ」タブで指定した MIB OID または式は、自動的に式に書き込まれます。

   a. 「トリガーしきい値」領域でリストから条件を選択し、MIB OID をフィルターする値を入力します。

   b. 「説明」フィールドに、トリガー式の分かりやすい説明を入力します。括弧書きの MIB 変数を説明に追加します。 イベントが発生すると、説明が AEL に表示されます。 例えば、次のようになります。

      ```
      CPU usage high (avgBusy5=)
      ```

   c. 基礎 EVAL ステートメントを説明に挿入するには、右括弧の内側にカーソルを置いて「**MIB** オブジェクトの追加 」をクリックし、指定した変数にナビゲートします。変数の現行値または前の値を評価するのか、値を SNMP 索引に解決するのかを指定して、「**OK**」をクリックします。 ステートメントが挿入されます。例:

      ```
      CPU usage high (avgBusy5=eval(text,"&SNMP.VALUE.sysName"))
      ```

   d. 「クリアしきい値」領域について、ステップ 7a から 7c を繰り返します。

8. 「保存」をクリックします。

関連概念:

12 ページの『ポーリング定義内のマルチバイト・データ』
中国語 (簡体字) などのマルチバイト文字を使用するドメインで Network Manager を実行している場合は、マルチバイト文字を処理するように Network Manager が構成されていることを確認してから、基本しきい値ポーリング定義または汎用しきい値ポーリング定義を構成する必要があります。

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

関連資料:

54 ページの『基本しきい値の式の例』
この基本しきい値の式の例を使用して、複雑な基本しきい値の式を構成する方法を理解してください。

221 ページの『付録 D. ポーリング定義の式の構文』
基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成する方法を理解するには、この情報を使用してください。

汎用しきい値ポーリング定義の変更

ポーリング定義エディターを使用して、汎用しきい値ポーリング定義を変更します。

ポーリング定義の一部の一般プロパティー、およびポーリング定義タイプに関連付けられているプロパティーを変更できます。ただし、ポーリング定義タイプを変更することはできません。

汎用しきい値ポーリング定義を変更するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 必要なポーリング定義をクリックします。 ポーリング定義は、汎用しきい値というポーリング定義タイプになっている必要があります。
3. ポーリング定義エディター の「一般」タブで、「一般プロパティー」の各フィールドに以下のように入力します。

   名前　ポーリング定義の固有の名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   タイプ

   このフィールドは使用不可です。このポーリング定義が、有効にされたポーリング・ポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。イベント ID に割り当てられる値は POLL-polldef です。

   イベント **ID**

   このフィールドは使用不可です。このポーリング定義が、有効にされたポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。「イベント **ID**」フィールドには、以下のようにデータが取り込まれます。

- これが新しいポーリング定義である場合、「イベント **ID**」フィールドには、値 POLL-*polldef* が取り込まれます。ここで、*polldef* は、現在のポーリング定義の名前です。
- 既存のポーリング定義をコピーすることによってポーリング定義を作成した場合、「イベント **ID**」には、コピーされたポーリング定義と同じ値が入ります。

注**:** 一部のより古いデフォルトのポーリングの「イベント **ID**」フィールドには、POLL-*polldef* 命名規則が使用されていないものがあります。

イベント重大度
重大度として有効な数値を指定します。重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。使用可能な重大度レベルのリストについては、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

説明　ポーリング定義の簡略説明を入力します。

4. 「クラス」タブをクリックします。「クラス」ツリーで、必要なクラスのチェック・ボックスを選択します。

   重要**:** すべてのクラスのチェック・マークを外すと、このポーリング定義を使用するポーリング・ポリシーで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。
5. オプション: 「インターフェース・フィルター」タブをクリックして、必須フィールドのフィルターを作成します。 「テーブル」フィールドには、interfaces テーブルのデータが事前に取り込まれています。
6. 「トリガーしきい値」タブをクリックします。以下のいずれかの方式を使用して、しきい値を指定する式を作成します。
   - 「基本」領域で、フィールドとオプションを使用して式を作成します。MIB ツリーから値を選択するには、「**MIB** ツリーを開く」 をクリックします。
   - 「拡張」領域で、オブジェクト照会言語 (OQL) で必要な **eval** ステートメントを入力します。
7. 生成されるイベントの AEL に表示されるメッセージを指定します。
   a. 「イベントの記述」フィールドに、メッセージを入力します。
   b. フィールドに MIB 変数を挿入するには、「**MIB** ツリーを開く」 をクリックします。現行の SNMP 値または前の SNMP 値か、SNMP 索引のいずれかを組み込むようにメッセージを設定して、「**OK**をクリックします。
8. 必須: 「しきい値のクリア」をクリックします。すべての汎用しきい値ポーリング定義には、クリアのしきい値が必要です。以下のいずれかの方式を使用して、しきい値を指定する式を作成します。

- 「基本」領域で、フィールドとオプションを使用して式を作成します。MIB ツリーから値を選択するには、「**MIB** ツリーを開く」 をクリックします。
- 「拡張」領域で、オブジェクト照会言語 (OQL) で必要な **eval** ステートメントを入力します。

ヒント**:** しきい値が手動でクリアされるようにしたい場合は、到達されることのないクリアのしきい値を作成してください。

9. 生成されるイベントの AEL に表示されるメッセージを指定します。
   a. 「イベントの記述」フィールドに、メッセージを入力します。
   b. フィールドに MIB 変数を挿入するには、「**MIB** ツリーを開く」 をクリックします。現行の SNMP 値または前の SNMP 値か、SNMP 索引のいずれかを組み込むようにメッセージを設定して、「**OK**をクリックします。
10. 「保存」をクリックし、次に「**OK**」をクリックします。

関連概念:

12 ページの『ポーリング定義内のマルチバイト・データ』
中国語 (簡体字) などのマルチバイト文字を使用するドメインで Network Manager を実行している場合は、マルチバイト文字を処理するように Network Manager が構成されていることを確認してから、基本しきい値ポーリング定義または汎用しきい値ポーリング定義を構成する必要があります。

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』
Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

54 ページの『汎用しきい値の式の例』
この汎用しきい値の式の例を使用して、複雑な汎用しきい値の式を構成する方法を理解してください。

221 ページの『付録 D. ポーリング定義の式の構文』
基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成する方法を理解するには、この情報を使用してください。

シャーシ ping およびインターフェース ping ポーリング定義の変更

ポーリング定義エディターを使用して、シャーシ ping およびインターフェース ping ポーリング定義タイプを変更します。

ポーリング定義の一部の一般プロパティー、およびポーリング定義タイプに関連付けられているプロパティーを変更できます。ただし、ポーリング定義タイプを変更することはできません。

シャーシ ping またはインターフェース ping ポーリング定義を変更するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 必要なポーリング定義をクリックします。 ポーリング定義は、シャーシ ping およびインターフェース ping というポーリング定義タイプになっている必要があります。
3. ポーリング定義エディター の「一般」タブで、「一般プロパティー」の各フィールドに以下のように入力します。

   名前　ポーリング定義の固有の名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   タイプ

   このフィールドは使用不可です。このポーリング定義が、有効にされたポーリング・ポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。イベント ID に割り当てられる値は POLL-polldef です。

   イベント **ID**

   このフィールドは使用不可です。このポーリング定義が、有効にされたポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。「イベント **ID**」フィールドには、以下のようにデータが取り込まれます。

   - これが新しいポーリング定義である場合、「イベント **ID**」フィールドには、値 POLL-*polldef* が取り込まれます。ここで、*polldef* は、現在のポーリング定義の名前です。
   - 既存のポーリング定義をコピーすることによってポーリング定義を作成した場合、「イベント **ID**」には、コピーされたポーリング定義と同じ値が入ります。

   注**:** 一部のより古いデフォルトのポーリングの「イベント **ID**」フィールドには、POLL-*polldef* 命名規則が使用されていないものがあります。

   イベント重大度

   重大度として有効な数値を指定します。重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。使用可能な重大度レベルのリストについては、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

   説明　ポーリング定義の簡略説明を入力します。
4. 「クラス」タブをクリックします。「クラス」ツリーで、必要なクラスのチェック・ボックスを選択します。

   重要**:** すべてのクラスのチェック・マークを外すと、このポーリング定義を使用するポーリング・ポリシーで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。

5. オプション: 「インターフェース・フィルター」タブをクリックして、必須フィールドのフィルターを作成します。 「テーブル」フィールドには、interfaces テーブルのデータが事前に取り込まれています。
6. 「**ping**」タブをクリックして、「**ping** プロパティー」フィールドに以下のように入力します。

   タイムアウト
   : ポーリング・プロセスが新しい ping パケットを送信する前にターゲット・デバイスからの応答を待つ時間をミリ秒単位で指定します。

   再試行
   : ポーリング・プロセスを中止する前にターゲット・デバイスの ping を試行する回数を指定します。「パケット・ロス」メトリック収集が有効になっている場合、ポーリング・プロセスは、応答が受信されたかどうかに関係なく、この数の ping パケットを送信します。

   ping メトリックの収集
   : 応答時間
     : デバイスが ping 要求に応答するために要した時間を収集するには、このボックスにチェック・マークを付けます。

     パケット・ロス
     : 失われたパケットに関するデータを収集するには、このボックスにチェック・マークを付けます。

   ペイロード・サイズ
   : ping 要求に使用する ICMP パケットのサイズを選択します。デフォルト (32 バイト) またはカスタム・サイズを選択します。この設定は、NcPollerSchema.cfg 構成ファイル内の IcmpData の値をオーバーライドします。

     注意:
     32 バイトより小さいサイズを使用すると、パケットがドロップされる場合があります。

7. 「保存」をクリックし、次に「**OK**」をクリックします。

関連概念:

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

リモート ping ポーリング定義およびリンク状態ポーリング定義の変更

ポーリング定義エディターを使用して、ポーリング定義タイプ (Cisco リモート ping、Juniper リモート ping、および SNMP リンク状態) を変更します。

ポーリング定義の一部の一般プロパティー、およびポーリング定義タイプに関連付けられているプロパティーを変更できます。ただし、ポーリング定義タイプを変更することはできません。

リモート ping ポーリング定義または SNMP リンク状態ポーリング定義を変更するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 必要なポーリング定義をクリックします。 ポーリング定義には、以下のポーリング定義タイプのいずれかがなければなりません。
   - Cisco リモート ping
   - Juniper リモート ping
   - SNMP リンク状態
3. ポーリング定義エディター の「一般」タブで、「一般プロパティー」の各フィールドに以下のように入力します。

   名前　ポーリング定義の固有の名前を指定します。使用できるのは、英数字、スペース、および下線のみです。

   タイプ

   このフィールドは使用不可です。このポーリング定義が、有効にされたポーリング・ポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。イベント ID に割り当てられる値は POLL-polldef です。

   イベント **ID**

   このフィールドは使用不可です。このポーリング定義が、有効にされたポリシーの一部として組み込まれると、ポーリング・エンジン (ncp_poller) は、このフィールドに自動的にデータを取り込みます。「イベント **ID**」フィールドには、以下のようにデータが取り込まれます。

   - これが新しいポーリング定義である場合、「イベント **ID**」フィールドには、値 POLL-*polldef* が取り込まれます。ここで、*polldef* は、現在のポーリング定義の名前です。
   - 既存のポーリング定義をコピーすることによってポーリング定義を作成した場合、「イベント **ID**」には、コピーされたポーリング定義と同じ値が入ります。

   注**:** 一部のより古いデフォルトのポーリングの「イベント **ID**」フィールドには、POLL-*polldef* 命名規則が使用されていないものがあります。

   イベント重大度

   重大度として有効な数値を指定します。重大度レベルは、IBM Tivoli Netcool/OMNIbus で定義される有効な重大度レベルに対応している必要があります。使用可能な重大度レベルのリストについては、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

   説明　ポーリング定義の簡略説明を入力します。
4. 「クラス」タブをクリックします。「クラス」ツリーで、必要なクラスのチェック・ボックスを選択します。

重要: すべてのクラスのチェック・マークを外すと、このポーリング定義を使用するポーリング・ポリシーで定義されているスコープに一致するすべてのデバイスが、システムによってポーリングされます。

5. オプション: 「インターフェース・フィルター」タブをクリックして、必須フィールドのフィルターを作成します。 「テーブル」フィールドには、interfaces テーブルのデータが事前に取り込まれています。
6. 「保存」をクリックし、次に「**OK**」をクリックします。

関連概念:

3 ページの『ポーリング・ポリシー・スコープ』
ポーリング・ポリシー・スコープでは、ポーリングするデバイスまたはデバイス・インターフェースを定義します。

関連タスク:

78 ページの『リンク状態ポーリングの構成』
ncp_poller プロセスが、既存のイベントがない場合にリンク状態ポーリングの初期状態を判別する方法を指定します。 **ncp_poller** プロセスは、最初のポーリングを使用してリンク状態ポーリングの初期状態を判別するか、またはクリア状態を想定します。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』
Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

カスタマイズされたポーリング定義の例

カスタマイズされたポーリング定義の例を使用して、既存の定義をコピーし、要件に合わせてカスタマイズします。

シナリオ

デバイスの処理能力が過度に使用されている場合、オペレーターにアラートを出すポーリング定義が必要です。平均 CPU 使用率が 80% を超えるとイベントが生成され、平均 CPU 使用率が 80% を下回るとイベントがクリアされるようにします。

必要な設定

『シナリオ』で説明されている要件に応答するポーリング定義を作成するには、以下のように設定します。

- 「新規ポーリング定義タイプの選択」パネルで、「**基本しきい値**」を選択します。
- ポーリング定義エディターの「ポーリング・データ」タブで、以下のように設定します。
  - 「単一 **OID**」が選択されていることを確認します。
  - 「単一 **OID**」領域は、以下の例に示されているように入力します。太字のテキストは、入力する必要がある項目を示しています。通常のテキストは、リストから選択する項目を示しています。

```
current   avgBusy5
```

- ポーリング定義エディターの「しきい値」タブで、以下のように設定します。
  - 「トリガーしきい値」領域は、以下の例に示されているように入力します。太字のテキストは、入力する必要がある項目を示しています。通常のテキストは、リストから選択する項目を示しています。斜体のテキストは、編集できない項目を示しています。

```
current   avgBusy5   >=   80
```

  - 「説明」フィールドは、以下のように入力します。
    1. 「CPU usage high (avgBusy5=)」と入力します。
    2. 右括弧の内側にカーソルを置いて「**MIB** オブジェクトの追加」 をクリックします。
    3. パス iso/org/dod/internet/private/enterprises/cisco/local/lsystem/avgBusy5 にナビゲートします。
    4. 「現行 **SNMP** 値」を選択して、「挿入」をクリックします。
  - 「クリアのしきい値」領域は、以下の例に示されているように入力します。太字のテキストは、入力する必要がある項目を示しています。通常のテキストは、リストから選択する項目を示しています。斜体のテキストは、編集できない項目を示しています。

```
current   avgBusy5   <   80
```

  - 「説明」フィールドは、以下のように入力します。
    1. 「CPU usage high (avgBusy5=)」と入力します。
    2. 右括弧の内側にカーソルを置いて「**MIB** オブジェクトの追加」 をクリックします。
    3. パス iso/org/dod/internet/private/enterprises/cisco/local/lsystem/avgBusy5 にナビゲートします。
    4. 「現行 **SNMP** 値」を選択して、「挿入」をクリックします。

関連概念:

12 ページの『ポーリング定義内のマルチバイト・データ』

中国語 (簡体字) などのマルチバイト文字を使用するドメインで Network Manager を実行している場合は、マルチバイト文字を処理するように Network Manager が構成されていることを確認してから、基本しきい値ポーリング定義または汎用しきい値ポーリング定義を構成する必要があります。

基本しきい値の式の例

この基本しきい値の式の例を使用して、複雑な基本しきい値の式を構成する方法を理解してください。

例: snmpInBandwidth

snmpInBandwidth ポーリング定義は、デフォルトのポーリング定義の一つです。このポーリング定義では、着信帯域幅使用率チェックが定義されています。着信帯域幅使用率が 40% を超えると、アラートが生成されます。以下の式は、eval ステートメントを使用してこの条件を定義する方法を示しています。この式は、「式」ラジオ・ボタンの下の「ポーリング・データ」タブで定義します。

```
((eval(long64,"&SNMP.DELTA.ifInOctets") / eval(long64,"&POLL.POLLINTERVAL"))
/(eval(long64,"&SNMP.VALUE.ifSpeed")))
*800
```

「しきい値」タブで、この式の値が 40 を超えるとトリガーするようにしきい値が設定されています。

関連概念:

12 ページの『ポーリング定義内のマルチバイト・データ』
中国語 (簡体字) などのマルチバイト文字を使用するドメインで Network Manager を実行している場合は、マルチバイト文字を処理するように Network Manager が構成されていることを確認してから、基本しきい値ポーリング定義または汎用しきい値ポーリング定義を構成する必要があります。

関連タスク:

44 ページの『基本しきい値ポーリング定義の変更』
ポーリング定義エディターを使用して、基本しきい値ポーリング定義を変更します。

27 ページの『基本しきい値ポーリング定義の作成』
MIB 変数に対して単純な式を実行したり、インターフェース・レベルのフィルターを使用するしきい値ポーリングを作成するには、基本しきい値ポーリング定義を作成します。

関連資料:

221 ページの『付録 D. ポーリング定義の式の構文』
基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成する方法を理解するには、この情報を使用してください。

汎用しきい値の式の例

この汎用しきい値の式の例を使用して、複雑な汎用しきい値の式を構成する方法を理解してください。

例: memoryPoll

memoryPoll ポーリング定義は、デフォルトのポーリング定義の 1 つです。このポーリング定義では、Cisco デバイスのメモリー・プール使用量チェックが定義されています。メモリー・プール使用率が 80% を超えると、アラートが生成されます。以下の式は、eval ステートメントを使用してこの条件を定義する方法を示しています。この式は、「拡張」ラジオ・ボタンの下の「トリガーしきい値」タブで定義します。

```
((eval(int,"&SNMP.VALUE.ciscoMemoryPoolValid") = 1)
AND
((eval(long64,"&SNMP.VALUE.ciscoMemoryPoolUsed")/
(eval(long64,"&SNMP.VALUE.ciscoMemoryPoolFree") +
eval(long64,"&SNMP.VALUE.ciscoMemoryPoolUsed")))*100
> 80))
```

関連概念:

12 ページの『ポーリング定義内のマルチバイト・データ』
中国語 (簡体字) などのマルチバイト文字を使用するドメインで Network Manager を実行している場合は、マルチバイト文字を処理するように Network Manager が構成されていることを確認してから、基本しきい値ポーリング定義または汎用しきい値ポーリング定義を構成する必要があります。

関連タスク:

46 ページの『汎用しきい値ポーリング定義の変更』
ポーリング定義エディターを使用して、汎用しきい値ポーリング定義を変更します。

30 ページの『汎用しきい値ポーリング定義の作成』
ポーリング定義エディターを使用して、新規の汎用しきい値ポーリング定義を作成します。

関連資料:

221 ページの『付録 D. ポーリング定義の式の構文』
基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成する方法を理解するには、この情報を使用してください。

第 6 章 ポーリング・ポリシーの削除

ポーリング・ポリシーが不要になった場合は削除します。

ポーリング・ポリシーを削除するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 「選択」列で、該当するポーリング・ポリシーを選択します。
3. 「削除  」をクリックします。
4. 「**OK**」をクリックして、削除を確認します。

選択したポーリング・ポリシーが削除されました。

© Copyright IBM Corp. 2006, 2016

第 7 章 ポーリング定義の削除

ポーリング定義が不要になった場合は削除します。

ポーリング定義を削除するには、以下のようにします。

1. 「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」をクリックします。
2. 「選択」列で、該当するポーリング定義を選択します。
3. 「削除」 をクリックします。
4. 「**OK**」をクリックして、削除を確認します。

選択したポーリング定義が削除されます。

© Copyright IBM Corp. 2006, 2016

第 8 章 適応ポーリングの管理

適応ポーリングは、ネットワークのイベントに動的に反応します。さまざまなネットワーク問題のシナリオを管理する適応ポーリングを作成することが可能です。

関連概念:

142 ページの『適応ポーリング・プラグイン』
ここでは、プラグインの前提条件、適応ポーリング・プラグインが activeEvent テーブルのフィールドにデータを取り込む方法、およびこのプラグインに関連する構成の詳細について説明します。 activeEvent テーブルは、NCMONITOR スキーマ内にあります。

関連タスク:

77 ページの『ポーリング・プログラムが読み取るイベントの数の構成』
ネットワーク・イベントの数が大幅に増加すると、ポーリング・プロセス **ncp_poller** の速度が低下することがあります。適応ポーリングのために、ポーリング・プログラムが読み取るイベントの数を定義して、イベント数の急増によるパフォーマンスへの影響を最小限に抑えることができます。

関連資料:

140 ページの『プラグインの説明』
ここでは、各イベント・ゲートウェイ・プラグインの仕組みについて説明します。

適応ポーリングのシナリオ

Network Manager は、主要なネットワーク問題のシナリオを自動的に処理するためのデフォルトの適応ポーリングを備えています。これらのデフォルトの適応ポーリングを使用すると、独自の適応ポーリングを作成する方法を理解できます。

デバイスが実際にダウンしていることの迅速な確認

このシナリオで説明する適応ポーリングを使用すると、デバイスが実際にダウンしたときに可能な限り迅速に判別できます。この適応ポーリングを活動化するには、ConfirmDeviceDown ポーリング・ポリシーを有効にします。

基本的な原理

標準のデフォルト・シャーシ ping ポーリング・ポリシーでは、現在のネットワーク・ドメイン内のすべてのシャーシ・デバイスが 2 分ごとにポーリングされます。さまざまな理由から、正常なデバイスがこのポーリング・ポリシーに応答せず、紛らわしい NmosPingFail イベントが **Active Event List (AEL)** に生成される場合があります。これらのイベントは、早くとも 2 分後の ping 応答に成功するまでクリアされません。その期間、この紛らわしいイベントによって、ネットワーク・オペレーターが不要なアクションを実行する可能性があります。

このシナリオで説明する適応ポーリングでは、ネットワーク・オペレーションによる不要なアクティビティーのリスクを回避するために、 NmosPingFail イベントが最初に生成されたすべてのデバイスに対して、高速 ping が実行されます。これら

© Copyright IBM Corp. 2006, 2016

のデバイスは、別のポーリング・ポリシーによって、3 分間の間 10 秒ごとに ping されます。これは、デバイスが応答したらすぐにイベントをクリアするためです。高速 ping を 3 分間実行しても NmosPingFail イベントが依然として発生しているデバイスは、実際にダウンしているものと見なされ、ping されなくなります。イベントが確実にすぐに使用できるので、ネットワーク・オペレーターがこれらのデバイスに対してアクションを実行することも、関連アクションを実行するための自動化を作成することもできます。

注**:** この 3 分間の ping は、NmosPingFail イベントが発生し合計値が 18 未満のデバイスをポリシー・スコープとして指定することで実行されます。合計値は、障害のあるデバイスがすべての高速ポーリングに応答しないと想定して計算されます。 1 分ごとに高速 ping ポーリングが 6 回実行されるので、3 分間では 18 回になり、依然として応答していないデバイスの合計値は 18 に達します。

高速 ping の値は、完全に構成できます。例えば、高速 ping ポーリングの間隔を (10 秒間隔ではなく) 20 秒に指定すれば、ネットワークの負荷を軽減できます。デバイスが実際にダウンしているかどうかを確認するためにより長い時間が必要な場合は、 (合計値を増やすことにより) 3 分より長い間、高速ポーリングを続けることもできます。

チェーニングされたポーリング・ポリシー

このシナリオで説明する適応ポーリングは、以下のチェーニングされたポーリング・ポリシーから構成されています。

デフォルト・シャーシ **ping (Default Chassis Ping)**
: このポーリング・ポリシーでは、ネットワーク内のすべてのデバイスが 2 分ごとに ping されます。デフォルトで有効になっています。

ConfirmDeviceDown
: このポーリング・ポリシーでは、デフォルト・シャーシ ping ポーリング・ポリシーに応答しないデバイスに対して、 10 秒間隔で高速 ping が実行されます。 ConfirmDeviceDown ポリシーは、ネットワーク・ビューに割り当てる必要があります。

  重要**:** ConfirmDeviceDown ポーリング・ポリシーはデフォルトで無効になっています。このシナリオで説明している適応ポーリングを活動化するには、このポーリング・ポリシーを有効にする必要があります。

シナリオのウォークスルー

以下のセクションでは、この適応ポーリングについて段階的に説明します。

1. デフォルト・シャーシ ping ポーリング・ポリシーでは、現在のネットワーク・ドメイン内のすべてのシャーシ・デバイスがポーリングされます。このポリシーのスコープを変更するには、関連するネットワーク・ビューとデバイス・フィルターを変更します。
2. デフォルト・シャーシ ping ポリシーに応答しないすべてのデバイスについて、NmosPingFail イベントが生成されます。

3. 関連する NmosPingFail イベントを持つデバイスは、初期 ping 失敗イベント・ネットワーク・ビューに自動的に追加されます。初期 ping 失敗イベント・ネットワーク・ビューは、以下のように定義されたフィルター済みネットワーク・ビューです。

```
EventId = NmosPingFail
Tally < 18
```

   初期 ping 失敗イベント・ネットワーク・ビューは、ネットワーク・ビューのネットワーク・ビュー・ツリーで「モニター・ビュー」 > 「初期 **ping** 失敗イベント」ノードを選択することで見つけることができます。ネットワーク・ビュー・ツリーのノードについて詳しくは、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

   高速ポーリングの期間を変更するには、 `Tally < 18` フィルター節を変更します。例えば、高速ポーリングの期間を 5 分間に延ばす場合は、この節を `Tally < 30` に変更します。この値は、10 秒のポーリング間隔に基づく計算で算出されます。つまり、1 分ごとに高速 ping ポーリングが 6 回実行されるので、5 分間では 30 になります。イベント・フィルター済みネットワーク・ビューの変更について詳しくは、*IBM Tivoli Network Manager IP Edition* ネットワーク可視化セットアップ・ガイドを参照してください。

4. ConfirmDeviceDown ポーリング・ポリシーでは、初期 ping 失敗イベント・ネットワーク・ビュー内のすべてのデバイスが 10 秒ごとにポーリングされます。デバイスが応答しないたびに、そのデバイスについて別の NmosPingFail が受信され、その NmosPingFail 合計値が増分されます。

   重要**:** このシナリオで説明している適応ポーリングを活動化するには、デフォルトでは、以下のアクションを実行する必要があります。

   - ConfirmDeviceDown ポーリング・ポリシーはデフォルトで無効になっています。このポーリング・ポリシーを有効にする必要があります。
   - デフォルトでは、ConfirmDeviceDown ポーリング・ポリシーにスコープはありません。初期 ping 失敗イベント・ネットワーク・ビューを、ConfirmDeviceDown ポーリング・ポリシーのスコープとして割り当てる必要があります。

   高速 ping ポーリングの頻度を変更するには、 ConfirmDeviceDown ポーリング・ポリシーを編集して、ポーリングの操作から操作までの間隔 (秒) を変更します。例えば、ポーリング頻度を 20 秒のポーリング間隔へと減少させる場合は (1 分あたり 6 回ではなく 3 回のポーリング)、ポーリング・ポリシー・エディターで ConfirmDeviceDown ポーリング・ポリシーを開き、「ポーリング間隔」の値に 20 を設定します。

   注**:** ポーリング・ポリシーの間隔を変更すると、同一の高速ポーリングの期間の合計値が変更されます。例えば、「ポーリング間隔」の値を 20 に変更すると、高速 ping ポーリングの頻度が 1 分あたり 3 ポーリングに減少します。この場合、3 分間の高速ポーリングの関連する合計値は 9 になります。この値は、20 秒のポーリング間隔に基づく計算で算出されます。つまり、1 分ごとに高速 ping ポーリングが 3 回実行されるので、3 分間では 9 になります。

5. デバイスに対する高速ポーリングは、以下のいずれかの方法で終了できます。

正常なデバイス

3 分間の高速ポーリング期間で、デバイスについて正常な ping 応答が受信されます。このデバイスの NmosPingFail イベントはクリアされ、次に **Active Event List (AEL)** から削除されます。このデバイスは、初期 ping 失敗イベント・ネットワーク・ビューから自動的に削除されるので、高速ポーリングは実行されなくなります。

障害のあるデバイス

デバイスは、3 分間の期間が終了するまで、ping ポーリングされ続けます。このデバイスのイベントの合計値が 18 に達すると、デバイスは初期 ping 失敗イベント・ネットワーク・ビューから自動的に削除されるので、高速ポーリングは実行されなくなります。 **Active Event List (AEL)** には、すぐに使用できるイベント、つまり合計値が 18 以上の NmosPingFail イベントが含まれます。

関連タスク:

13 ページの『第 2 章 ポーリングの有効化と無効化』

Network Manager ポーリングをアクティブにするには、ポーリング・ポリシーを有効にする必要があります。ネットワーク・エンティティーがネットワーク上にない場合は、そのエンティティーをポーリングするポーリング・ポリシーを無効にします。

37 ページの『ポーリング・ポリシーの変更』

ポーリング・ポリシー・エディターを使用して、既存のポーリング・ポリシーの設定を変更します。

67 ページの『適応ポーリングの作成』

ネットワークのイベントにシステムが動的に反応できるようにするには、適応ポーリングを作成します。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』

Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

211 ページの『付録 B. デフォルトのポーリング定義』

Network Manager IP Editionには、一般的なポーリング要件に対応したさまざまなデフォルトのポーリング定義が用意されています。

しきい値違反の迅速な確認

このシナリオで定義する適応ポーリングを使用すると、通常はネットワーク・デバイスへの影響を最小限に抑えるために低速 SNMP ポーリングを実行し、しきい値違反が実際に発生したときに高速ポーリングを実行できます。この適応ポーリングを活動化するには、ConfirmHighDiscardRate ポーリング・ポリシーを有効にします。

基本的な原理

標準の HighDiscardRate ポーリング・ポリシーでは、現在のネットワーク・ドメイン内のすべてのルーターに対して 30 分ごとに SNMP しきい値ポーリングが実行され、パケット破棄率 (%) が 5% を超えているルーター・インターフェースがないかどうかが判別されます。ルーター上のいずれかのインターフェースでしきい値

を超えていることがポーリング・ポリシーで検出されると、そのルーターについて POLL-HighDiscardRate イベントが生成されます。正常なルーター・インターフェースでも、ビジー状態になり、パケットの除去が必要になる場合があります。これにより、いずれかの 30 分の間隔で 5% のしきい値違反が発生し、紛らわしい HighDiscardRate イベントが **Active Event List (AEL)** に生成される場合があります。これらのイベントは、早くとも 30 分後に発生する POLL-HighDiscardRate ポーリング・ポリシーの応答が成功するまでクリアされません。その期間、この紛らわしいイベントによって、ネットワーク・オペレーターが不要なアクションを実行する可能性があります。

このシナリオで説明する適応ポーリングでは、ネットワーク・オペレーションによる不要なアクティビティーのリスクを回避するために、POLL-HighDiscardRate イベントが最初に生成されたすべてのデバイスに対して、高速 SNMP ポーリングが実行されます。これらのデバイスは、別の SNMP ポーリング・ポリシーによって 5 分ごとにポーリングされます。これは、デバイスのすべてのインターフェースでパケット破棄率 (%) が 5% のしきい値内に収まったという応答があり次第、イベントをクリアするためです。POLL-HighDiscardRate イベントが引き続き発生しているデバイスは、そのインターフェースの 1 つ以上に障害があるものと確認されます。イベントが確実にすぐに使用できるので、ネットワーク・オペレーターがこれらのデバイスに対してアクションを実行することも、関連アクションを実行するための自動化を作成することもできます。

高速ポーリングの値は、完全に構成できます。例えば、高速 SNMP ポーリングの間隔を (5 分間隔ではなく) 10 分に指定すれば、ネットワークの負荷を軽減できます。

チェーニングされたポーリング・ポリシー

このシナリオで説明する適応ポーリングは、以下のチェーニングされたポーリング・ポリシーから構成されています。

HighDiscardRate

このポーリング・ポリシーでは、デバイスのインターフェースでドロップされているパケット数が、処理中のパケット総数に対する最小パーセントを超えていないかどうかが判別されます。このポリシーでは、この情報が 30 分ごとにポーリングされます。

ConfirmHighDiscardRate

このポーリング・ポリシーでは、パケット破棄率 5% のしきい値を違反したインターフェースを 1 つ以上持つデバイスに対して、高速 SNMP ポーリングが 5 分間隔で実行されます。ConfirmHighDiscardRate ポリシー・スコープは、ネットワーク・ビューに割り当てる必要があります。

重要**:** ConfirmHighDiscardRate ポーリング・ポリシーは、デフォルトでは無効になっています。このシナリオで説明している適応ポーリングを活動化するには、このポーリング・ポリシーを有効にする必要があります。

シナリオのウォークスルー

以下のセクションでは、この適応ポーリングについて段階的に説明します。

1. HighDiscardRate SNMP ポーリング・ポリシーでは、現在のネットワーク・ドメイン内のすべてのルーターに対して SNMP しきい値ポーリングが 30 分ごとに実行されます。このポリシーのスコープを変更するには、関連するネットワーク・ビューとデバイス・フィルターを変更します。
2. POLL-HighDiscardRate イベントは、パケット破棄率 5% のしきい値を違反したインターフェースを 1 つ以上持つすべてのデバイスに対して生成されます。
3. 関連する POLL-HighDiscardRate イベントを持つデバイスは、HighDiscardRate ネットワーク・ビューに対して 1 つ以上のインターフェース・イベントを持つデバイスに自動的に追加されます。このネットワーク・ビューは、以下のように定義されたフィルター済みネットワーク・ビューです。

   ```
   EventId = POLL-HighDiscardRate
   ```

   HighDiscardRate ネットワーク・ビューについて 1 つ以上のインターフェース・イベントを持つデバイスは、ネットワーク・ビューのネットワーク・ビュー・ツリーで「モニター・ビュー」 > 「**HighDiscardRate** について **1** つ以上のインターフェース・イベントを持つデバイス」ノードを選択することで見つけることができます。ネットワーク・ビュー・ツリーのノードについて詳しくは、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。
4. ConfirmHighDiscardRate ポーリング・ポリシーでは、HighDiscardRate ネットワーク・ビューに対して 1 つ以上のインターフェース・イベントを持つデバイス内のすべてのデバイスが 5 分ごとにポーリングされます。

   重要**:** このシナリオで説明している適応ポーリングを活動化するには、デフォルトでは、以下のアクションを実行する必要があります。

   - ConfirmHighDiscardRate ポーリング・ポリシーは、デフォルトでは無効になっています。このポーリング・ポリシーを有効にする必要があります。
   - デフォルトでは、ConfirmHighDiscardRate ポーリング・ポリシーにスコープはありません。 HighDiscardRate ネットワーク・ビューに対して 1 つ以上のインターフェース・イベントを持つデバイスを、ConfirmHighDiscardRate ポーリング・ポリシーのスコープとして割り当てる必要があります。

   重要**:** ConfirmHighDiscardRate ポーリング・ポリシーは、デフォルトでは無効になっています。このシナリオで説明している適応ポーリングを活動化するには、このポーリング・ポリシーを有効にする必要があります。

   高速 SNMP しきい値ポーリングの頻度を変更するには、ConfirmHighDiscardRate ポーリング・ポリシーを編集して、ポーリングの操作から操作までの間隔 (秒) を変更します。例えば、ポーリング頻度を 10 分のポーリング間隔へと減少させる場合は、ポーリング・ポリシー・エディターで ConfirmHighDiscardRate ポーリング・ポリシーを開き、「ポーリング間隔」の値に 600 を設定します。
5. デバイスに対する高速 SNMP しきい値ポーリングは、以下のいずれかの方法で終了できます。

正常なデバイス

デバイスのすべてのインターフェースは、高速ポーリングに対して、しきい値である 5% 以内のパケット破棄率で応答します。このデバイスの POLL-HighDiscardRate イベントはクリアされ、次に **Active Event List (AEL)** から削除されます。このデバイスは、HighDiscardRate ネットワーク・ビューについて 1 つ以上のインターフェース・イベントを持つデバイスから自動的に削除されるので、高速ポーリングは実行されなくなります。

障害のあるデバイス

デバイスの 1 つ以上のインターフェースは、高速 SNMP ポーリングに対して、 5% のパケット破棄率を違反していると応答し続けます。POLL-HighDiscardRate イベントは **Active Event List (AEL)** に残り、合計値は増加し続けます。**Active Event List (AEL)** には、すぐに使用できる POLL-HighDiscardRate イベントが含まれます。

関連タスク:

13 ページの『第 2 章 ポーリングの有効化と無効化』
Network Manager ポーリングをアクティブにするには、ポーリング・ポリシーを有効にする必要があります。ネットワーク・エンティティーがネットワーク上にない場合は、そのエンティティーをポーリングするポーリング・ポリシーを無効にします。

37 ページの『ポーリング・ポリシーの変更』
ポーリング・ポリシー・エディターを使用して、既存のポーリング・ポリシーの設定を変更します。

『適応ポーリングの作成』
ネットワークのイベントにシステムが動的に反応できるようにするには、適応ポーリングを作成します。

関連資料:

205 ページの『付録 A. デフォルトのポーリング・ポリシー』
Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

211 ページの『付録 B. デフォルトのポーリング定義』
Network Manager IP Editionには、一般的なポーリング要件に対応したさまざまなデフォルトのポーリング定義が用意されています。

適応ポーリングの作成

ネットワークのイベントにシステムが動的に反応できるようにするには、適応ポーリングを作成します。

適応ポーリングを作成する前に、適応ポーリングから利点が得られるネットワーク状態を最初に識別する必要があります。例えば、Network Manager でデフォルトで提供される適応ポーリングでは、以下のいずれかを行うために、選択した一連のデバイスに対して高速ポーリングが実行されます。

- ICMP ping が失敗したデバイスが実際にダウンしているかどうかを確認する。
- デバイスのしきい値が実際に違反されているかどうかを確認する。

2 つのポーリング・ポリシーのチェーニングを作成する適応ポーリングを作成するには、以下のようにします。 Network Manager でデフォルトで提供される適応ポーリングからの例を「デバイスのダウンの確認」列と「しきい値違反の確認」列に示します。

手順	デバイスのダウンの確認	しきい値違反の確認
1. ネットワーク内のデバイスのエラー状態を取得する既存のポーリング・ポリシーを識別します。	使用するポーリング・ポリシー: デフォルト・シャーシ **ping** ネットワーク・ドメイン内のすべてのデバイスに対して 2 分ごとに ping ポーリングを実行します。	使用するポーリング・ポリシー: **HighDiscardRate** デバイスのインターフェースでドロップされているパケット数が、処理中のパケット総数に対する最小パーセントを超えていないかどうかを判別します。この情報を 30 分ごとにポーリングします。
2. 前のステップで識別したポーリングで生成されるイベントに基づいてデバイスをフィルタリングするイベント・フィルター済みネットワーク・ビューを作成します。 通常、このネットワーク・ビュー内のデバイスには関連するエラー状態がありますが、そのエラー状態は、より頻繁なポーリングによって詳細に診断することが可能です。 イベント・フィルター済みネットワーク・ビューの作成方法については、*IBM Tivoli Network Manager IP Edition* ネットワーク可視化セットアップ・ガイドを参照してください。	ネットワーク・ビュー: 「モニター・ビュー」 > 「初期 **ping** 失敗イベント」 NmosPingFail イベントが発生し合計値が指定値未満のデバイスがすべて含まれます。NmosPingFail イベントは、デフォルト・シャーシ ping ポーリング・ポリシーが失敗したイベントで発生します。	ネットワーク・ビュー: 「モニター・ビュー」 > 「**HighDiscardRate** について **1** つ以上のインターフェース・イベントを持つデバイス」 デバイスのインターフェースでドロップされているパケット数が、処理中のパケット総数に対する最小パーセントを超えていないかどうかを判別します。

手順	デバイスのダウンの確認	しきい値違反の確認
3. 前のステップで作成したネットワーク・ビューをスコープとして持ち、そのネットワーク・ビュー内のデバイスをより頻繁にポーリングするポーリング・ポリシーを作成します。 これらのデバイスをより頻繁にポーリングする目的は、以降のアクションの準備として問題を詳細に診断するためです。	ポーリング・ポリシー: **ConfirmDeviceDown** 頻度の高いポーリングの目的: 実際にダウンしているデバイスを識別するために、初期 ping 失敗イベント・ネットワーク・ビュー内のデバイスに対する ping ポーリングを 10 秒のポーリング間隔へと高速化します。正常なデバイスでは、この頻度の高いポーリングに対して成功の応答が戻され、そのイベントはクリアされます。	ポーリング・ポリシー: **ConfirmHighDiscardRate** 頻度の高いポーリングの目的: 対応する前によりタイムリーな情報を得るために、HighDiscardRate について 1 つ以上のインターフェース・イベントを持つデバイス・ネットワーク・ビュー内のデバイスに対するポーリングを高速化します。このポーリング・ポリシーでは、POLL_HighDiscardRate イベントが引き続き生成され、デバイスの問題が確認されるか、またはエラー・イベントをクリアする解決済みイベントが発行され、関連するデバイスが HighDiscardRate ビューから削除されます。

ステップ 2 では、「デバイスのダウンの確認」列の「初期 *ping* 失敗イベント」ネットワーク・ビューには、合計値を使用して実装された終了基準が含まれています。イベントの合計値が指定値を超えると、関連するデバイスが自動的にネットワーク・ビューから削除されます。これは、限られた時間で高速ポーリングを実行して、そのデバイスの状態を設定するときに役立ちます。状態が設定されると、デバイスをビューから削除できます。例えば、このネットワーク・ビューのデフォルト設定の場合は、 ping ポーリングが失敗したデバイスに対して、高速ポーリングを 3 分間のみ実行します。 3 分経っても関連する NmosPingFail イベントがデバイスにある場合、そのデバイスは、ダウンしているものと確認されます。

追加のネットワーク・ビューとポーリング・ポリシーを作成し、ネットワーク状態に応じて必要な診断を実行するように適切にチェーニングすることにより、3 つ以上のポーリング・ポリシーをチェーニングすることも可能です。

関連概念:

61 ページの『デバイスが実際にダウンしていることの迅速な確認』
このシナリオで説明する適応ポーリングを使用すると、デバイスが実際にダウンしたときに可能な限り迅速に判別できます。この適応ポーリングを活動化するには、ConfirmDeviceDown ポーリング・ポリシーを有効にします。

64 ページの『しきい値違反の迅速な確認』
このシナリオで定義する適応ポーリングを使用すると、通常はネットワーク・デバイスへの影響を最小限に抑えるために低速 SNMP ポーリングを実行し、しきい値違反が実際に発生したときに高速ポーリングを実行できます。この適応ポーリングを活動化するには、ConfirmHighDiscardRate ポーリング・ポリシーを有効にします。

関連タスク:

15 ページの『第 3 章 ポーリングの作成』
既存のデフォルト・ポーリング・ポリシーおよび定義が要件に合わない場合に、ポーリングを作成します。既存のポーリングおよびデフォルト・ポーリングのコピーをカスタマイズするか、新規のポーリングを最初から作成してください。

第 9 章 ネットワーク・ポーリングの管理

コマンド行インターフェースを使用して、幅広いポーリング管理タスクを実行します (複数のポーリング・プログラム機能、ネットワーク・ドメイン間のネットワーク・ポーリングのコピー、ネットワーク・ポーリングの中断、ポーリングの有効化と無効化、ポーリング状況の取得、およびポーリングのリフレッシュなど)。

SNMP v2 または v3 が使用されている場合は、SNMP ヘルパーを構成して GetBulk 操作を使用することもできます。 GetBulk 操作を使用すると、ポーリングの効率性が向上します。詳しくは、*IBM Tivoli Network Manager IP Edition* インストールと構成ガイド を参照してください。

ポーリングの管理

ポーリング・ポリシーを管理するには、コマンド行インターフェースを使用します。

SNMP 資格情報のチェックを行わないことによる ncp_poller 始動速度の向上

Fix Pack 3

ncp_poller 始動時の SNMP アクセス資格情報のテストを無効にすると、始動時間が短縮されます。

ポーリング対象のデバイスのアクセス資格情報が正確であることが確実である場合は、ポーリング・プロセス **ncp_poller** を、始動時に資格情報をチェックしないよう構成することができます。

1. ファイル `NCHOME/etc/precision/NcPollerSchema.DOMAIN.cfg` をバックアップして編集します。
2. DiscoverInitialAccess オプションの値を定義する行を見つけます。
3. **ncp_poller** プロセスの始動時に毎回 SNMP 資格情報をチェックする場合は、DiscoverInitialAccess の値を `1` に設定したままにします。
4. SNMP アクセス資格情報の初期チェックをオフにするには、DiscoverInitialAccess の値を `0` に設定します。この設定でも、ポーリングの障害が発生する場合には、**ncp_poller** プロセスは特定のデバイスの SNMP 資格情報をテストします。
5. **ncp_poller** プロセスを再始動します。

© Copyright IBM Corp. 2006, 2016

ポーリング状況の取得

ポーリング・ポリシーの状況を表示するには、**itnm_poller.pl** スクリプトを使用します。

コマンド行インターフェース (CLI) で提供されるドメインには、domainMgr NCIM テーブル内の項目がなければなりません。

itnm_poller.pl スクリプトについて詳しくは、*IBM Tivoli Network Manager IP Edition* 管理ガイドを参照してください。

1. `$NCHOME/precision/scripts/perl/scripts` ディレクトリーに移動して、**itnm_poller.pl** スクリプトを探します。
2. ポーリング・ポリシーの状況を取得するには、**itnm_poller.pl** スクリプト・プログラムを実行します。 以下の例は、すべてのポーリング・ポリシーの状況を取得する方法を示しています。

   ```
   ncp_perl itnm_poller.pl -domain NCOMS -status all
   ```

   `-status realtime` または `-status static` を指定して、リアルタイムのポーリング・ポリシーの状況、または静的なポーリング・ポリシーの状況のみを取得することもできます。

ポーリングの有効化と無効化

個々のポーリング・ポリシーを有効または無効にするには、itnm_poller.pl スクリプトを使用します。

コマンド行インターフェース (CLI) で提供されるドメインには、domainMgr NCIM テーブル内の項目がなければなりません。

itnm_poller.pl スクリプトについて詳しくは、*IBM Tivoli Network Manager IP Edition* 管理ガイドを参照してください。

1. `$NCHOME/precision/scripts/perl/scripts` ディレクトリーに移動して、itnm_poller.pl スクリプトを探します。
2. さまざまなポーリング・ポリシーの状況を取得するには、次の例に示すように、itnm_poller.pl スクリプト・プログラムを実行します。

   ```
   ncp_perl itnm_poller.pl -domain NCOMS -status all
   ```

   コマンドの出力では、ポーリング・ポリシーごとに ID が表示されます。有効または無効にするポーリング・ポリシーの ID を書き留めてください。
3. 個々のポーリング・ポリシーを有効または無効にするには、ポーリング・ポリシー ID をパラメーターとして指定して、itnm_poller.pl スクリプト・プログラムを実行します。 以下の例は、ポリシー ID が 10 のポーリング・ポリシーを有効にする方法を示しています。

   ```
   ncp_perl itnm_poller.pl -domain NCOMS -enable 10
   ```

   以下の例は、ポリシー ID が 15 のポーリング・ポリシーを無効にする方法を示しています。

   ```
   ncp_perl itnm_poller.pl -domain NCOMS -disable 15
   ```

ポーリングのリフレッシュ

ポーリング・ポリシー構成とそのエンティティー・リストをリフレッシュするには、 itnm_poller.pl スクリプトを使用します。

コマンド行インターフェース (CLI) で提供されるドメインには、NCIM トポロジー・データベース domainMgr テーブル内の項目がなければなりません。

itnm_poller.pl スクリプトについて詳しくは、*IBM Tivoli Network Manager IP Edition* 管理ガイドを参照してください。

1. $NCHOME/precision/scripts/perl/scripts ディレクトリーに移動して、itnm_poller.pl スクリプトを探します。
2. さまざまなポーリング・ポリシーの状況を取得するには、次の例に示すように、itnm_poller.pl スクリプト・プログラムを実行します。

   ```
   ncp_perl itnm_poller.pl -domain NCOMS -status all
   ```

   コマンドの出力では、ポーリング・ポリシーごとに ID が表示されます。有効または無効にするポーリング・ポリシーの ID を書き留めてください。
3. itnm_poller.pl スクリプト・プログラムを実行して、単一のポーリング・ポリシーまたは複数のポーリング・ポリシーをリフレッシュします。 以下の例は、ポリシー ID が 10 のポーリング・ポリシーをリフレッシュする方法を示しています。

   ```
   ncp_perl itnm_poller.pl -domain NCOMS -refresh 10
   ```

   以下の例は、すべてのポーリング・ポリシーをリフレッシュする方法を示しています。

   ```
   ncp_perl itnm_poller.pl -domain NCOMS -refresh all
   ```

ドメイン間のポーリングのコピー

get_policies.pl プログラムを使用して、ポーリング・ポリシーをあるネットワーク・ドメインから別のネットワーク・ドメインに、またはファイルとドメイン間でコピーします。

-to か -from のいずれかのオプションが付いたドメイン名を提供する場合、NCIM および NCMONITOR データベースへの接続が必要です。接続は、DbLogins.*DOMAIN*.cfg ファイル、またはドメイン固有のファイルが見つからない場合は、DbLogins.cfg ファイルの設定に基づいて作成されます。

コマンド行インターフェース (CLI) で提供されるドメインには、domainMgr NCIM テーブル内の項目がなければなりません。

1. NCHOME/precision/scripts/perl/scripts ディレクトリーに移動して、get_policies.pl プログラムを見つけます。
2. get_policies.pl プログラムを実行して、ポーリング・ポリシーをコピーします。 以下の表は、考えられるアクションと CLI での入力内容について説明します。

アクション	**CLI** への入力
すべてのポーリング・ポリシーをあるドメインから別のドメインに直接コピーします。	ncp_perl get_policies.pl -from domain=*SOURCE* -to domain=*DESTINATION* -ncim_password *NCIM_password* -ncmonitor_password *NCMONITOR_password*
選択したポーリング・ポリシーのみ、あるドメインから別のドメインに直接コピーします。	ncp_perl get_policies.pl -from domain=*SOURCE* -to domain=*DESTINATION* -policy "*policy_name1*" -policy "*policy_name2*"
すべてのポーリング・ポリシーをドメインから **XML** ファイルにコピーします。	ncp_perl get_policies.pl -from domain=*DOMAIN_name* -to file=*filename*.xml -password *NCIM_password*
すべてのポーリング・ポリシーを **XML** ファイルからドメインにコピーします。	ncp_perl get_policies.pl -from file=*filename*.xml -to domain=*DOMAIN_name*
選択したポーリング・ポリシーのみドメインからファイルにコピーします。	ncp_perl get_policies.pl -from domain=*DOMAIN_name* -to file=*filename*.xml -policy "*policy_name1*" -policy "*policy_name2*"

ポーリング中断オプション

デバイスまたはコンポーネントを非管理状態に設定すると、そのデバイスまたはコンポーネントを対象にした Network Manager のネットワーク・ポーリングは中断します。

デバイスまたはコンポーネントが非管理状態になっていると、そのデバイスまたはコンポーネントを対象にした Network Manager のポーリングは実行されず、イベントも生成されません。さらに、そのデバイスのイベントに関する根本原因分析 (RCA) も実行されません。

制限事項

デバイスを「非管理」状態に設定しても、その影響を受けるのは Network Manager のポーリングだけであり、その他のポーリングによってデバイスやコンポーネントから情報を取得し、該当する場合にアラートを生成する操作が停止するわけではありません。ただしその場合でも、他のソースからのアラートには非管理状態のタグが付くので、そのタグによって、アラートの対象のデバイスまたはコンポーネントが保守状態になっていることを確認できます。

非管理状態のデバイスの設定の詳細については、*IBM Tivoli Network Manager IP Edition* ディスカバリー・ガイドを参照してください。

ポーリングを中断するためのオプション

デバイスまたはコンポーネントのアクティブなポーリング操作を中断するには、以下のようなオプションがあります。

デバイスまたはコンポーネントを非管理状態に設定する方法

以下のオプションがあります。

- 「ネットワーク・ビュー」または「ホップ・ビュー」を使用します。

- **Unmanage node** ツールを使用します。

  デバイスを非管理状態に設定すると、そのデバイスに含まれているすべてのコンポーネントも非管理状態になります。デバイスに含まれている個々のコンポーネントを非管理状態に設定すると、そのコンポーネントとそのコンポーネントに含まれているすべてのコンポーネントだけが非管理状態になり、デバイス自体は管理状態のまま残ります。コンポーネントの管理状態/非管理状態を設定できるのは、そのコンポーネントが含まれているデバイスが管理状態になっている場合に限られます。さらに、シャーシに関連した管理インターフェースを非管理状態に設定しても、デバイス全体のポーリングが中断するわけではありません。

特定のインターフェース・タイプを永久に非管理状態に設定する方法
: これにより、指定されたインターフェース・タイプが Network Manager によってポーリングされなくなります。さらに、新しいデバイスをディスカバーしてトポロジーに追加した後に、最初からそのデバイスをポーリングの対象外にするように設定することもできます。その設定は、`TagManagedEntities.stch` ファイルと NCIM データベース表によって行います。

「ネットワーク・ビュー」と「ホップ・ビュー」の詳細については、*IBM Tivoli Network Manager IP Edition* ネットワーク可視化セットアップ・ガイドを参照してください。デバイスやコンポーネントを非管理状態に設定する方法の詳細については、*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイドを参照してください。

ポーリング帯域幅の調整

ポーリング・エンジン ncp_poller が転送するデータの量と頻度を構成できます。ネットワーク輻輳を回避したり、大量のポーリング・イベントが同時に発生した場合の影響を低減したりするために、ポーリング帯域幅の調整が必要な場合もあります。

ポーリング・データ・キューのサイズの調整

Fix Pack 4

NCPOLLDATA データベースに書き込むためのポーリング・データのキューが大きくなりすぎると、データベースの問題やポーリング・ロードが大きいことが原因で、ポーリング・プログラムがメモリー不足になることがあります。この問題をモニターして回避するために、キューのサイズについて制限を設定できます。この制限は、バッチの数として定義されます。制限を超過した場合、ログおよびアクティブ・イベント・リスト (AEL) でアラートが表示されます。その場合、ポーリング・プログラムは、キューに入れられたデータを破棄して、キューのサイズを減らします。破棄されたデータは、別のファイルに書き込まれるため、そのデータを取得してデータベースに書き込むことができます。

ポーリング・プログラムのデバッグ・レベルを 4 に設定します。

1. 制限を設定するには、ポーリング・プログラムの $NCHOME/etc/precision/NcPollerSchema.cfg ファイルで、**PollDataQueueLimit** パラメーターを適切なバッチ数に設定します。 以下の例では、キューに入れられるデータの制限を 50 バッチに設定する方法を示します。

   ```
   update config.properties set PollDataQueueLimit = 50;
   ```

2. $NCHOME/log/precision 内のポーリング・プログラムのログ・ファイルをモニターし、AEL でメッセージおよびアラートをモニターします。

PollDataQueueLimit パラメーターによって指定された制限をキューが超過した場合、以下のアクションが実行されます。

- AEL で、アラートが表示されます。例:

  ```
  ItnmPollerPolicyDataQueueFull: ポーリング・プログラム NCOMS。
  ポーリング・データ・キューがキャパシティーを超過しました。データをファイルにオフロードしています
  ```

- メッセージがログに書き込まれます。例:

  ```
  2013-04-19T12:37:58 [CDataQueue::ProcessValue] キュー・サイズがしきい値を
  超過しました。
  データを破棄しています: policyId:122:templateId:39:monitoredInstanceId:2212
  :monitoredObjectId:3:pollTime:1387232947:
  tdwTime:1131216222944000:errorCode:111:value:0
  ```

- ポーリング・データがキューから破棄され、$NCHOME/log/precision/ncp_poller.*poller name*.*domain*.polldata ファイル (例えば、ncp_poller.*poller name*.*domain*.data) に書き込まれます。データは、SQL INSERT ステートメントとして書き込まれます。例えば、次のようになります。

  ```
  INSERT INTO pollData( MONITOREDOBJECTID, MONITOREDINSTID, POLLTIME, TDWTIME,
  ERRORCODE, VALUE )  VALUES ( 3, 2828, 1387232686, 1131216172446000, 100, 714 );
  INSERT INTO pollData( MONITOREDOBJECTID, MONITOREDINSTID, POLLTIME, TDWTIME,
  ERRORCODE, VALUE )  VALUES ( 3, 2829, 1387232686, 1131216172446000, 100, 715 );
  ```

制限を超過した場合、以下のようにします。

- ポーリング・データ・キューのサイズを解決します。例えば、追加のポーリング・プログラムを作成したり、データベース管理者に連絡したりします。
- AEL からアラートをクリアします。
- .polldata ファイルから NCPOLLDATA データベースにデータを書き込みます。次に、.polldata ファイルをクリアします。このファイルは、自動的にクリアもローテートもされません。

関連タスク:

81 ページの『追加のポーリング・プログラムの設定』
ネットワークの負荷の処理に 1 つのポーリング・プログラムだけでは不十分な場合は、Network Manager サーバーで追加のポーリング・プログラムを設定します。ポーリング・プログラムを追加するには、**ncp_poller** プロセスの新しいインスタンスを登録して作成します。

90 ページの『ポーリング・プログラムのキャパシティーのモニター』
ポーリング・プログラムで問題が発生しないようにするには、ポーリング・プログラムのメトリックをコマンド行インターフェースに出力してモニターします。メトリックは、棒グラフとして表示されます。メトリックは、ポーリング・プログラムがキャパシティーに達しようとしているのかどうか (例えば、不十分なデータベース・パフォーマンスにより、ポーリング・プログラムで遅延が発生しているかどう

か) を示します。

ポーリング・プログラムが読み取るイベントの数の構成

Fix Pack 4

ネットワーク・イベントの数が大幅に増加すると、ポーリング・プロセス **ncp_poller** の速度が低下することがあります。適応ポーリングのために、ポーリング・プログラムが読み取るイベントの数を定義して、イベント数の急増によるパフォーマンスへの影響を最小限に抑えることができます。

適応 **ncp_poller** プロセスは、NCMONITOR.activeEvent テーブルからメモリーにイベントを一定の間隔で読み込んで、ポーリング・イベントの初期状態を判別します。ポーリング・プログラムが読み取るイベントの数に上限を設定できます。上限を超えるイベントはメモリーに読み込まれません。イベントの数が再び上限を下回ると、ポーリング・プログラムはイベントの読み取りを再開します。

1. `$NCHOME/etc/precision/NcPollerSchema.cfg` ファイルをバックアップしてから、編集します。
2. 既存の行を編集またはアンコメントするか、新しい行を追加して、以下の行を指定します。

   ```
   update config.properties set EventThrottle = number_of_events
   ```

   ここで、*number_of_events* は、ポーリング・プログラムが読み取るイベントの最大数です。この行が存在しない場合、ポーリング・プログラムが処理を試みるイベントの数は制限されません。
3. ファイルを保存して閉じます。

関連概念:

142 ページの『適応ポーリング・プラグイン』
ここでは、プラグインの前提条件、適応ポーリング・プラグインが activeEvent テーブルのフィールドにデータを取り込む方法、およびこのプラグインに関連する構成の詳細について説明します。 activeEvent テーブルは、NCMONITOR スキーマ内にあります。

関連タスク:

61 ページの『第 8 章 適応ポーリングの管理』
適応ポーリングは、ネットワークのイベントに動的に反応します。さまざまなネットワーク問題のシナリオを管理する適応ポーリングを作成することが可能です。

関連資料:

140 ページの『プラグインの説明』
ここでは、各イベント・ゲートウェイ・プラグインの仕組みについて説明します。

ポーリング・ポリシーのメンバーシップ・サイズをチェックする間隔の変更

すべてのポーリング・ポリシーに対するポーリング・ポリシーのメンバーシップ・サイズをチェックする間隔を変更することができます。

ポーリング・ポリシーのメンバーシップ・サイズをチェックする間隔を変更する前に、以下の点を検討する必要があります。

- 間隔を長くすると、チェック間の時間が長くなるためにネットワーク上の負荷が減ります。
- 間隔を短くすると、変更を反映して ncp_poller を最新の状態にする頻度が高くなります (特にネットワーク・ビューのメンバーシップ・サイズ)。

1. 構成ファイル $NCHOME/etc/precision/NcPollerSchema.cfgを編集します。
2. ファイルの最後に、以下の行を追加します。

   ```
   update config.properties set PolicyUpdateInterval= update_interval;
   ```

   update_interval は、ポーリング・ポリシーのメンバーシップ・サイズをチェックする秒単位の間隔です。デフォルト値は 30 秒です。
3. ポーリング・エンジン ncp_poller を再始動します。

ポーリング・ポリシーに対するネットワーク・ビューの更新の有効化と無効化

デフォルトでは、すべてのポーリング・ポリシーでネットワーク・ビューの更新が有効になります。安定したネットワーク・ビューをポーリングしている場合は、ネットワーク・ビューの更新を無効にすることができます。これにより、更新に関連するパフォーマンスの問題を回避できます。

1. 構成ファイル $NCHOME/etc/precision/NcPollerSchema.cfgを編集します。
2. ファイルの最後に、以下の行を追加します。

   ```
   update config.properties set UpdateNetworkViewCache = enable_or_disable_update;
   ```

   enable_or_disable_update は、1 (有効) または 0 (無効) のいずれかです。デフォルト値は 1 (有効) です。
3. ポーリング・エンジン ncp_poller を再始動します。

リンク状態ポーリングの構成

ncp_poller プロセスが、既存のイベントがない場合にリンク状態ポーリングの初期状態を判別する方法を指定します。 **ncp_poller** プロセスは、最初のポーリングを使用してリンク状態ポーリングの初期状態を判別するか、またはクリア状態を想定します。

1. NCHOME/etc/precision/NcPollerSchema.cfg ファイルを編集します。
2. config.properties セクションを見つけます。
3. **UseFirstPollForInitialState** の値を編集します。
   - 初期状態をクリアに設定する場合は、0 に設定します。
   - 初期状態を最初のポーリングによって判別する場合は、1 に設定します。

関連概念:

6 ページの『リンク状態ポーリング』
リンク状態ポーリングは、ifOperStatus および ifAdminStatus インターフェース MIB 変数における状況の変化をモニターします。これら変数の値がポーリング間隔の間に変化すると、イベントが発生します。

関連タスク:

50 ページの『リモート ping ポーリング定義およびリンク状態ポーリング定義の変更』
ポーリング定義エディターを使用して、ポーリング定義タイプ (Cisco リモート ping、Juniper リモート ping、および SNMP リンク状態) を変更します。

SNMP しきい値ポーリングの構成

Fix Pack 5

カウンター変数 (帯域幅変数など) の差異を測定する SNMP しきい値ポーリングは、モニター対象デバイスが再始動する場合に誤った値を記録する可能性があります。

デバイスの再始動中に実行されるポーリングはすべて破棄するように、ポーリング・プロセスを構成できます。

このようにポーリング・プロセスを構成するには、以下の手順を実行します。

1. Network Manager コア・コンポーネントがインストールされているサーバー上で以下のファイルをバックアップおよび編集します。`$NCHOME/etc/precision/NcPollerSchema.cfg`
2. 以下の行を追加します。

   ```
   update config.properties property set DropWrappedPolls = 1
   ```
3. **ncp_poller** プロセスを再始動します。

複数のポーリング・プログラムの管理

ネットワークをポーリングするために複数のポーリング・プログラムが必要な場合は、Network Manager を設定して複数のポーリング・プログラム機能を管理することができます。ポーリング・プログラムの追加や削除、およびポーリング・プログラム ID を使用して特定のポーリング・プログラムにポリシーを関連付けることができます。

複数のポーリング・プログラムの概要

より多くのデバイスにポーリングを拡張するため、およびパフォーマンス上の理由のため、複数のポーリング・プログラムを使用できます。

ポーリング・プログラム・プロセスは、ネットワーク・デバイスのポーリングと、データのプルーニングやキャッシュの更新などの管理用タスクの実行という 2 つの機能を備えています。単一のポーリング・プログラムが日常的に多数のデバイスにポーリングしている場合は、ポーリング用に複数のポーリング・プログラムを使用することを検討してください。

注: 複数のポーリング・プログラムを使用し、それぞれ別に構成したい場合、ポーリング・プログラム構成ファイル NcPollerSchema.cfg のドメイン固有のバージョンおよびポーリング・プログラム固有のバージョンを作成できます。ドメイン固有のバージョンを作成するには、ファイル名にドメイン名を追加します。例えば、NcPollerSchema.*DOMAIN*.cfg です。ポーリング・プログラム固有のバージョンを作成するには、ファイル名にポーリング・プログラム名とドメイン名を追加します。例えば、NcPollerSchema*POLLERNAME*.*DOMAIN*.cfg です。ポーリング・プログラム・インスタンスは、使用可能な構成ファイルのうちで、最も限定的な構成ファイルを使用します。

別のポーリング・プログラムを使用して管理機能を実行することで、パフォーマンスの問題を回避することもできます。-admin オプションを指定して 1 つのポーリング・プログラムのみを開始し、そのポーリング・プログラムはネットワーク・デバイスのポーリングに使用しないようにします。他のポーリング・プログラムは -noadmin オプションを指定して開始し、ネットワーク・ポーリングに使用します。フェイルオーバーを使用する場合は、同じ配置のポーリング・プログラムをバックアップ・サーバーに複製します。

各ポーリング・プログラムは、管理者がポリシーをポーリング・プログラムに関連付けるときに使用できる名前で登録されます。

制約事項: 複数のサーバーにわたってポーリングを分散することは、サポートされていません。Network Manager サーバー上のすべてのポーリング・プログラムを開始します。

IBM Tivoli Monitoring の Network Manager ヘルス・ビューを使用して、ポーリング・エンジンをモニターできます。

詳しくは、「*IBM Tivoli Monitoring for Tivoli Network Manager IP Edition* ユーザー・ガイド」を参照してください。

モニター・ポリシー・エディター

複数のポーリング・プログラムがデプロイされると、管理者はどのポーリング・プログラムがポリシーによって使用されるかを制御します。デフォルトでは、各ポリシーは Network Manager サーバー上のデフォルトのポーリング・プログラムを使用します。管理者は、登録されたポーリング・プログラムのリストから別のポーリング・プログラムを選択できます。

リアルタイム **MIB** グラフ

リアルタイム・ポーリングに使用されるポーリング・プログラムは、管理者によって定義されます。

追加のポーリング・プログラムの設定

ネットワークの負荷の処理に 1 つのポーリング・プログラムだけでは不十分な場合は、Network Manager サーバーで追加のポーリング・プログラムを設定します。ポーリング・プログラムを追加するには、**ncp_poller** プロセスの新しいインスタンスを登録して作成します。

デフォルトでは、製品のインストール中にポーリング・プログラムが 1 つ自動的にインストールされます。このデフォルトのポーリング・プログラムに名前はありません。

1. ポーリング・プログラムを登録するには、$ITNMHOME/bin/ncp_poller コマンドを実行します。 例えば、POLL2344 というポーリング・プログラムを NCOMS ドメインに登録するには、以下のコマンドを実行します。

   ```
   ncp_poller -domain NCOMS -register -name POLL2344
   ```

2. $NCHOME/etc/precision/CtrlServices.cfg ファイルに、ポーリング・プログラムの新しいインスタンスを作成します。
   a. **ncp_poller** プロセスのエントリーのコピーを作成します。
   b. このコピーにある **serviceName** パラメーターを、作成したばかりのポーリング・プログラムの名前に変更します。
   c. 必要に応じて他の **ncp_poller** オプションを変更します。

      Fix Pack 3 必ずいずれかのポーリング・プログラムを -admin オプション (重要な管理機能を実行するようにポーリング・プログラムを構成するオプション) で開始し、それ以外のポーリング・プログラムを -noadmin で開始してください。CtrlServices.cfg ファイルの **ncp_poller** 項目の例については、『例』を参照してください。

3. 指定したドメインで、**ncp_ctrl** プロセスを再始動します。 新しいポーリング・プログラムを含め、実行中のすべての **ncp_*** プロセスが再始動されます。

例

以下の例では、SNMP 照会に使用できる snmpPoller というポーリング・プログラムを NCOMS ドメインに作成します。

```
insert into services.inTray
(
    serviceName,
    binaryName,
    servicePath,
    domainName,
    argList,
    dependsOn,
    retryCount
)
values
(
    "snmpPoller",
    "ncp_poller",
    "$PRECISION_HOME/platform/$PLATFORM/bin",
    "$PRECISION_DOMAIN",
    [ "-domain" , "$PRECISION_DOMAIN" , "-latency" , "60000", "-debug", "0",
"-messagelevel", "warn", "-name", "snmpPoller"],
    [ "nco_p_ncpmonitor", "ncp_g_event" ],
    5
);
```

以下の例では、ping 照会に使用できる PingPoller というポーリング・プログラムを作成します。

```
insert into services.inTray
(
    serviceName,
    binaryName,
    servicePath,
    domainName,
    argList,
    dependsOn,
    retryCount
)
values
(
    "PingPoller",
    "ncp_poller",
    "$PRECISION_HOME/platform/$PLATFORM/bin",
    "$PRECISION_DOMAIN",
    [ "-domain" , "$PRECISION_DOMAIN" , "-latency" , "60000", "-debug", "0",
"-messagelevel", "warn", "-name", "PingPoller"],
    [ "nco_p_ncpmonitor", "ncp_g_event" ],
    5
);
```

Fix Pack 3 以下の例では、ping 照会に使用できるポーリング・プログラムを NCOMS ドメインに作成します。このポーリング・プログラムは管理ポーリング・プログラムとして設定されます。

```
insert into services.inTray
(
    serviceName,
    binaryName,
    servicePath,
    domainName,
    argList,
    dependsOn,
    retryCount
)
values
(
    "PingPoller",
    "ncp_poller",
    "$PRECISION_HOME/platform/$PLATFORM/bin",
    "$PRECISION_DOMAIN",
    [ "-domain" , "$PRECISION_DOMAIN" , "-latency" , "60000", "-debug", "0",
"-messagelevel", "warn", "-name", "PingPoller" "-admin"],
    [ "nco_p_ncpmonitor", "ncp_g_event" ],
    5
);
```

Fix Pack 3 以下の例では、SNMP 照会に使用できるポーリング・プログラムを NCOMS ドメインに作成します。このポーリング・プログラムは、前記の例に示したポーリング・プログラムと同じ環境で実行するため、-noadmin として実行するように設定されます。

```
insert into services.inTray
(
    serviceName,
    binaryName,
    servicePath,
    domainName,
    argList,
    dependsOn,
```

```
    retryCount
)
values
(
    "snmpPoller",
    "ncp_poller",
    "$PRECISION_HOME/platform/$PLATFORM/bin",
    "$PRECISION_DOMAIN",
    [ "-domain" , "$PRECISION_DOMAIN" , "-latency" , "60000", "-debug", "0",
"-messagelevel", "warn", "-name", "snmpPoller" "-noadmin"
 ],
    [ "nco_p_ncpmonitor", "ncp_g_event" ],
    5
);
```

ポーリング・プログラムの追加後に以下を実行します。

1. 製品を再始動します。
2. ポーリング定義を編集します。
3. ポーリング・ポリシーの「次のポーリング・プログラム・インスタンスへの割り当て」フィールドを設定することにより、ポーリングを割り当てます。

関連タスク:

43 ページの『ポーリング定義の変更』
既存のポーリング定義を変更して、ポーリング要件に合わせてカスタマイズします。ポーリング定義の変更は、ポーリング定義エディターで行います。実行するステップは、ポーリング定義タイプ ごとに異なります。

37 ページの『ポーリング・ポリシーの変更』
ポーリング・ポリシー・エディターを使用して、既存のポーリング・ポリシーの設定を変更します。

MIB グラフのポーリング・プログラムの割り当て

追加のポーリング・プログラムを定義済みで、MIB グラフを処理するためにポーリング・プログラムを割り当てる場合は、プロパティー・ファイルを編集します。

ファイルを編集する前に、MIB グラフに使用するポーリング・プログラムを作成しておく必要があります。

MIB グラフのポーリング・プログラムを割り当てるには、以下の手順を実行します。

1. ファイル ITNMHOME/profiles/TIPProfile/etc/tnm/tnm.properties を開きます。
2. **tnm.graph.poller** パラメーターの値を DEFAULT_POLLER から必要なポーリング・プログラムの名前に変更します。
3. ファイルを保存して閉じます。

ポーリング・プログラムの除去

ネットワークのモニターにポーリング・プログラムが使用されていない場合には、モニター・システムからポーリング・プログラムを除去できます。

モニター・システムからポーリング・プログラムを除去するには、コマンド行からポーリング・プログラムを登録解除します。ポーリング・プログラムがモニター・ポリシー・エディターから削除されます。

モニター・システムからポーリング・プログラムを除去するには、以下のタスクを完了します。

1. ポーリング・プログラムを削除する前に、削除するポーリング・プログラムに関連付けられた各アクティブ・ポリシーを編集します。モニター・ポリシー・エディターから、別の有効なポーリング・プログラムを使用するようにポリシーを編集します。
2. 実行中のすべての Network Manager プロセスを停止します。
3. Network Manager サーバーから、以下のコマンドを実行します。

   ```
   ncp_poller -deregister -domain [domain_name] -name [poller_name]
   ```

   ポーリング・プログラムに関連付けられているアクティブ・ポリシーがない場合は、そのポーリング・プログラムは登録解除されます。ポーリング・プログラムに関連付けられているアクティブ・ポリシーがまだ存在する場合は、スクリプトはそのポーリング・プログラムを登録解除しません。
4. ポーリング・プログラムに関連付けられているアクティブ・ポリシーがまだ存在する場合は、ポリシーを別のポーリング・プログラムに割り当てるか、`- force` オプションを使用してスクリプトを再度実行します。`- force` オプションを使用してスクリプトを実行した場合、ポーリング・プログラムに関連付けられているポーリング・ポリシーも削除されます。
5. `CtrlServices.cfg` ファイルを編集して、削除したポーリング・プログラムの **ncp_poller** エントリーを削除するか、そのコメントを外します。
6. Network Manager プロセスを再始動します。

ヒストリカル・ポーリング・データの管理

ヒストリカル・ポーリング・データを管理するには、コマンド行インターフェースを使用します。

ストレージ容量の考慮事項

ここでは、ストレージ容量に関連する問題、ガイドライン、および計算方法について説明します。ストレージ容量の要件を理解しておくと、Network Manager がヒストリカル・ポーリング・データのストレージ限度の増加に対応できるかどうかを判別するために役立ちます。

考慮事項として重要なのは、保管する必要があるデータの量およびデータが収集される速度です。デフォルトでは、Network Manager のポーリング・プログラムは、最新の 500 万データベース行を維持するように、プルーニング・ポリシーを保守します。この値は、2,000 万データベース行以上に増やすことができますが、

2,000 万行を超えると、レポート作成のパフォーマンスに影響が生じ、データベースの調整と保守が必要になることがあります。

「データ・ストレージの持続速度」という用語は、データをデータベース内に挿入できる平均速度を意味します。以下に示す例での計算には、日次速度が使用されます。これを上回る速度を使用することも可能ですが、パイプラインのどこかで必ず発生する短時間の中断 (ネットワークの中断、あるいはピーク負荷、再始動、割り込みを原因とするデータベースの速度低下など) を考慮して余裕を持たせることが重要です。

データ・ストレージの持続速度は、以下のような多くの要因に依存します。

- ポーリング・エンティティーの数
- ポーリングされるメトリックの数
- ポーリングの頻度
- ポリシーの数
- ポーリング・プログラムの数
- データベース行の数
- データベースのパフォーマンス (データ・ストレージの速度が大きい場合、データベースが低速であると影響が発生します)

一般に、ローカル NCPOLLDATA ヒストリカル・ポーリング・データ・データベース (これ以降、NCPOLLDATA データベースと呼びます) への挿入速度を 1 日当たり 700 万データベース行より大きくすることは、長期的には適切ではありません。単一のポーリング・ポリシーは、1 日当たり約 100 万データベース行を保管します。推奨される合計行数の範囲内で、複数のポリシーおよび複数のポーリング・プログラムを使用できます。Network Manager を使用して、ポーリング・プログラムの数を増やすことができます。NCPOLLDATA データベースへの過度の影響を回避するために、以下のガイドラインに従ってください。

- データの保管に使用するポーリング・プログラムは、3 つ以下にします。
- Network Manager のポーリング・プログラム 1 つにつき、ITM エージェント・インスタンスを 1 つ作成します。

Tivoli Data Warehouse へのスループットは、すべてのポーリング・プログラムの合計で、1 日当たり最大で 700 万データベース行です。この限度を超えると、障害に対する許容度が低くなり、データの損失が発生することがあります。これを上回る速度を達成することも可能ですが、ネットワーク・リンクおよび転送プロセスでエラー状態が発生する可能性があることを考慮する必要があります。Network Manager は、転送障害が短期間であれば、障害を許容し、データの損失なしで復旧します。この期間は、データ速度、ディスク・スペース、転送障害に関連する時間の長さなどの要因の組み合わせによって異なります。転送障害の期間が長いほど、システムの復旧にかかる時間も長くなります。

ヒストリカル・ポーリング・データのストレージ容量のガイドライン

ここでは、Network Manager がヒストリカル・ポーリング・データのストレージ限度の増加に対応できるかどうかを判別するために役立つストレージ容量の計算に関連するガイドラインについて説明します。

NCPOLLDATA ヒストリカル・ポーリング・データ・データベースまたは Tivoli Data Warehouse に格納されるデータベース行の発生率を計算するには、以下のガイドラインに従います。

- デバイスに関連付けられているヒストリカル・ポーリング・データの場合 (SNMP ポーリングまたは ICMP ポーリング):

  ```
  Number of device level database rows per day =
  number_of_devices * 60 * 24/polling_freq
  ```

  値の意味は次のとおりです。

  - *number_of_devices* — SNMP ポーリングまたは ICMP ポーリングの実行対象デバイスの数を指定します。
  - `60 * 24` — 1 日分の分数を指定します。
  - *polling_freq* — ポーリング対象デバイスに関連付けられているヒストリカル・ポーリング・データのポーリング頻度を分単位で指定します。SNMP ポーリングでのデバイスに関連付けられているヒストリカル・ポーリング・データの例としては、メモリー使用状況や CPU サイクルがあります。

    ICMP ポーリングの場合、ヒストリカル・ポーリング・データ項目 (アップ/ダウン状況、応答時間、およびパケット・ロスを確認するための ping) を指定できます。

- ネットワーク・インターフェースに関連付けられているヒストリカル・ポーリング・データの場合 (SNMP ポーリングまたは ICMP ポーリング):

  ```
  Number of network interface level database rows per day =
  number_of_network_interfaces * number_of_data_points *
  60 * 24/polling_freq
  ```

  値の意味は次のとおりです。

  - *number_of_network_interfaces* — SNMP ポーリングまたは ICMP ポーリングの実行対象ネットワーク・インターフェースの数を指定します。
  - *number_of_data_points* — データ・ポイントの数を指定します。SNMP ポーリングの実行対象ネットワーク・インターフェースに関連付けられている各ヒストリカル・ポーリング・データのデータ・ポイントを割り当てます。例えば、帯域幅および `ifInDiscards` のヒストリカル・ポーリング・データが必要な場合は、*number_of_data_points* に `2` を設定します。

    ICMP ポーリングの場合、ヒストリカル・ポーリング・データ項目 (アップ/ダウン状況、応答時間、およびパケット・ロスを確認するための ping) を指定できます。アップ/ダウン状況のヒストリカル・ポーリング・データを確認するための ping を選択する場合は、1 つのデータ・ポイントをカウントします。アップ/ダウン状況と応答時間のヒストリカル・ポーリング・データを確認するための ping を選択する場合は、2 つのデータ・ポイントをカウントします。3 つのヒストリカル・ポーリング・データをすべて選択する場合は、3 つのデータ・ポイントを選択します。例えば、アップ/ダウン状況と応

答時間を確認するための ping のヒストリカル・ポーリング・データが必要な場合は、*number_of_data_points* に 2 を設定します。

- 60 * 24 — 1 日分の分数を指定します。
- *polling_freq* — ポーリング対象ネットワーク・インターフェースに関連付けられているヒストリカル・ポーリング・データのポーリング頻度を分単位で指定します。ネットワーク・インターフェースに関連付けられているヒストリカル・ポーリング・データの例には、帯域幅および MIB オブジェクト (SNMP MIB II インターフェース表の ifInDiscards と ifInErrors など) があります。

  ICMP ポーリングの場合、ヒストリカル・ポーリング・データ項目 (アップ/ダウン状況、応答時間、およびパケット・ロスを確認するための ping) を指定できます。

ストレージ容量の計算の例

ここでは、Network Manager がヒストリカル・ポーリング・データのストレージ限度の増加に対応できるかどうかを判別するために役立つストレージ容量の計算方法について説明します。

ユーザーは、平均で 5 個のネットワーク・インターフェースがある 1000 個のデバイスをポーリングして、以下のヒストリカル・ポーリング・データを以下のポーリング間隔で取得します。

デバイス・レベル
: ヒストリカル・ポーリング・データ: メモリー使用状況、5 分間隔

インターフェース・レベル
: ヒストリカル・ポーリング・データおよびポーリング頻度 (SNMP): ifInDiscards、10 分間隔

  ヒストリカル・ポーリング・データおよびポーリング頻度 (SNMP): ifInErrors、10 分間隔

  ヒストリカル・ポーリング・データおよびポーリング頻度 (SNMP): 帯域幅、10 分間隔

  ヒストリカル・ポーリング・データおよびポーリング頻度 (ICMP): アップ/ダウン状況および応答時間の ping、5 分間隔

ユーザーは、これらのデバイス・レベルおよびインターフェース・レベルのポーリング要件に基づいて、前述のガイドラインに従ってデータベース行の日次速度を計算します。

注: この例の SNMP ポーリングでは、ifInDiscards、ifInErrors、および帯域幅のヒストリカル・ポーリング・データに対して、3 つのデータ・ポイントのカウントが指定されています。ICMP ポーリングでは、アップ/ダウン状況および応答時間のヒストリカル・ポーリング・データに対して、2 つのデータ・ポイントのカウントが指定されています。

```
Number of device level database rows per day (SNMP) =
1000 devices * 60 * 24/5 polls per day = 288,000

Number of network interface level database rows per day (SNMP) =
1000 * 5 interfaces * 3 data points * 60 * 24/10 polls per day = 2,160,000
```

```
Number of ICMP database rows per day =
1000 devices * 5 interfaces * 2 data points * 60 * 24/5 polls per day  = 2,880,000

Total database rows per day = 5,328,000
```

上記の例は、1 日当たりの合計データベース行数として 5,328,000 を示しています。これは、1 日当たり 700 万データベース行の上限ガイダンスの範囲内です。したがって、この例は、ヒストリカル・ポーリング・データのストレージ限度を 2,100 万データベース行または 2,200 万データベース行に増加させた後で、約 4 日分の生データを維持するシナリオを示しています。

ヒストリカル・ポーリング・データのストレージ限度の増加

ヒストリカル・ポーリング・データのストレージ限度を増加できます。 これにより、パフォーマンス・レポートに大量のヒストリカル・ポーリング・データを表示できるようになります。

Network Manager パフォーマンス・レポートは、操作性に重点を置いて、短期の診断用に設計されています。本格的なパフォーマンス管理製品 (Tivoli Netcool Performance Manager など) では、標準的な機能として、データ・ストレージの最適化や長期間にわたる日常的なデータ収集などを行います。これらの機能はキャパシティー・プランニングや規制に準拠したレポート作成のために必要ですが、Network Manager ヒストリカル・レポートの目標はこれとは異なります。ローカル・データベースまたは Tivoli Data Warehouse 内に、最大で 2,000 万データベース行を保管できます。ご使用のハードウェアとデータベースのパフォーマンスによっては、2,000 万データベース行を超えると、ストレージとレポート作成のパフォーマンスが低下することがあります。また、場合によっては、ストレージ速度とテーブル・サイズの増加に応じて、データベース管理者がデータベース上で統計とトランザクション・ログの保守を定期的に最適化および実行する必要が生じることもあります。

注: Network Manager は、デフォルトで、NCIM トポロジー・データベース内のデータベース・スキーマを使用して NCPOLLDATA データベースを実装します。

ヒストリカル・ポーリング・データのストレージ限度は、デフォルトでは少なめの 500 万データベース行に設定されています。この値は増加できますが、増加することでパフォーマンス・レポートのパフォーマンスが低下するおそれがあります。

データ・プルーニングは、行がデータベースから常時削除されることがないように、バッチ方式で実行されます。行数が限度を超え、超過分が 10,000 行になると、行数が限度に到達するまで行が削除されます。

ヒストリカル・ポーリング・データのストレージ限度を増加するには、以下の手順を実行します。

1. 以下のファイルを見つけます。
   - UNIX `$NCHOME/etc/precision/NcPollerSchema.cfg`
   - Windows `%NCHOME%¥etc¥precision¥NcPollerSchema.cfg`
2. 以下の行を見つけます。

   ```
   insert into config.pruning (MAXPOLLDATAROWS) values (5000000);
   ```

3. MAXPOLLDATAROWS の値を変更します。

   重要: この数値を増加すると、レポートのパフォーマンスが低下するおそれがあります。

ヒストリカル・ポーリング・データの削除

データベース内のヒストリカル・ポーリング・データが多すぎる場合は、ポーリング・データのプルーニング・スクリプトを使用して、ポーリング・データのサブセットを削除できます。

行数やデータの存続期間などの基準によってヒストリカル・ポーリング・データのサブセットを定義し、データを削除することができます。

ポーリング・データのプルーニング・スクリプトは IBM Tivoli Monitoring for IBM Tivoli Network Manager IP Edition を使用してインストールされます。

ポーリング・データのプルーニング・スクリプトを実行するには、以下のタスクを実行します。

1. IBM Tivoli Monitoring for IBM Tivoli Network Manager IP Edition がインストールされているサーバー上で、NCHOME/precision/products/tnm/bin ディレクトリーに移動します。
2. 以下のようなコマンド行を使用して、スクリプトを開始します。

   - UNIX itnm_polldata_pruning [-help] -username *user* -password *password* {-days *days* [-domain *domain*] [-policy *policy*]} | {-rows *rows*} [-prompt]
   - Windows itnm_polldata_pruning.bat [-help] -username *user* -password *password* {-days *days* [-domain *domain*] [-policy *policy*]} | {-rows *rows*} [-prompt

   コマンド行パラメーターは、以下の表にリストされています。

表 6. *itnm_polldata_pruning* スクリプトのコマンド行パラメーター

パラメーター	説明
-days *days*	データベース内にヒストリカル・ポーリング・データを保持する日数。ヒストリカル・ポーリング・データを削除するには、-days または -rows パラメーターを使用する必要があります。
-domain *domain*	オプション。指定されたドメインを持つ行を削除します。-days と共に使用します。
-help	スクリプトのヘルプを表示します。
-password *password*	ヒストリカル・ポーリング・データを保管するデータベースへのアクセス権限を持つユーザーのパスワード。
-policy *policy*	オプション。指定されたポリシー名を持つ行を削除します。-days と共に使用します。

表 6. *itnm_polldata_pruning* スクリプトのコマンド行パラメーター (続き)

パラメーター	説明
-prompt	オプション。ヒストリカル・ポーリング・データを削除する前に、ユーザーに対してプロンプトを出します。
-rows *rows*	削除する行数。最も古いデータから削除を開始します。-days パラメーターと共に使用することはできません。

ポーリング・プログラムのキャパシティーのモニター

Fix Pack 4

ポーリング・プログラムで問題が発生しないようにするには、ポーリング・プログラムのメトリックをコマンド行インターフェースに出力してモニターします。メトリックは、棒グラフとして表示されます。メトリックは、ポーリング・プログラムがキャパシティーに達しようとしているのかどうか (例えば、不十分なデータベース・パフォーマンスにより、ポーリング・プログラムで遅延が発生しているかどうか) を示します。

デフォルトでは、メトリックは、2 分ごとに NCHOME/log/precision 内のトレースに書き込まれます。ポーリング・プログラムごとに 1 つのトレースが存在します。トレースのファイル名の形式は、デフォルトのポーリング・プログラムの場合は ncp_poller.SnmpPoller.*domain*.metrics、その他のすべてのポーリング・プログラムの場合は ncp_poller.SnmpPoller.*pollername*.*domain*.metrics になります。例えば、ncp_poller.SnmpPoller.Poller23507.NCOMS.metrics のようになります。以下の表で、メトリックについて説明します。

表 7. ポーリング・プログラムのメトリック

メトリック	測定の対象	測定の単位 (各棒グラフの **y** 軸の単位)
Health	ポリシー・サイクル時にポーリングされたデバイスのパーセンテージ。この値が 100% の場合、ポーリング・プログラムは正しく機能しています。値が 100% 未満の場合は、ポーリング間隔中に一部のデバイスがポーリングされていません。ポーリング・プログラムは、ポリシーのロードに対応できていません。	%

表 7. ポーリング・プログラムのメトリック (続き)

メトリック	測定の対象	測定の単位 (各棒グラフの y 軸の単位)
Memory	ポーリング・プログラムが使用しているメモリー。メモリー使用量は、ディスカバーされるデバイスが多ければ多いほど、また有効なポリシーが多ければ多いほど増加します。	MB
BatchQueueSize	スレッドを待機しているバッチの数。	カウント
PollDataQueueSize	NCPOLLDATA データベースに対してキューに入れられた INSERT ステートメントの数。ポーリング・プログラムがポーリング・データを正常に保管しているかどうかを示します。	カウント
PollDataRowCount	ポーリング・プログラムがデータを除去した後の ncpolldata.polldata テーブル内の行数。このテーブルには、ヒストリカル・ポーリング・データが保管されます。デフォルトのポーリング・プログラムのみがデータを除去します。このメトリックが役に立つのは、ヒストリカル・ポーリングを使用している場合のみです。	カウント

-metrics オプションと -status オプションを同時に指定してスクリプトを実行しないでください。

棒グラフを出力する端末の幅が 140 文字以上であることを確認してください。そうでない場合、行が折り返されるため、棒グラフが正しく表示されません。

例に示されているように itnm_poller.pl スクリプトを実行します。 以下の例では、最新のタイム・スタンプからデフォルトの 4 時間の期間にわたる、NCOMS ドメインのデフォルト・ポーリング・プログラムのグラフを表示する方法を示します。

```
ncp_perl itnm_poller.pl -domain NCOMS -metrics
```

以下の例では、過去 12 時間にわたる、特定のポーリング・プログラムの同じドメインについて、スクリプトを実行する方法を示します。

```
ncp_perl itnm_poller.pl -domain NCOMS -poller Poller23507 -metrics -window 12
```

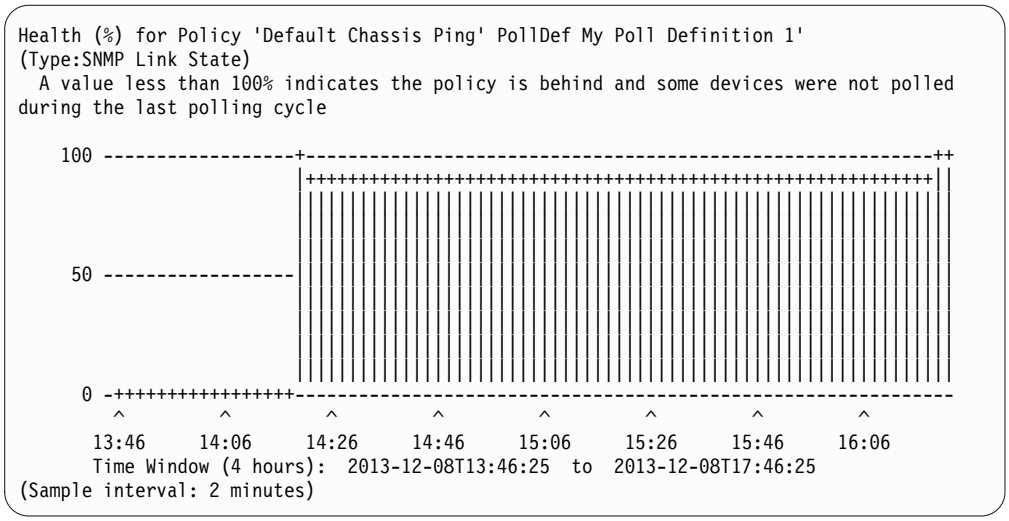
以下の例では、特定のタイム・スタンプからの 8 時間の期間にわたる、同じドメインおよびポーリング・プログラムについて、スクリプトを実行する方法を示します。

```
ncp_perl itnm_poller.pl -domain NCOMS -poller Poller23507 -metrics
-timestamp 2013-12-10T17:30:36 -window 8
```

y 軸のスケールによく注意してください。長期間 (例えば、24 時間) にわたって平坦に見える棒グラフは、グラフの期間を短く (例えば、4 時間に) して表示すると、値に差があることが示されることがあります。

例

以下の例では、正常性メトリックのサンプル・グラフを示します。



以下の表で問題および考えられる原因を確認して、必要に応じてアクションを実行します。

表 *8.* ポーリング・プログラムのメトリックでの問題の考えられる原因

メトリック	問題	考えられる原因	アクション
Health	値が一貫して 100% を下回っている。	ポーリング・プログラムの開始後、または変更情報を MODEL データベースから受信した場合、パーセンテージが一時的に 100% を下回ることがあります。	• ポーリング・ポリシーを変更して、ポーリング間隔を大きくします。 • ポーリング・プログラムをさらに追加します。

表 *8.* ポーリング・プログラムのメトリックでの問題の考えられる原因 (続き)

メトリック	問題	考えられる原因	アクション
Memory	メモリーが無制限に増大する。	データベースへの接続が失われました。あるいは、ポーリング・ロードが大きすぎて維持できないか、データ・ストレージ・レートが大きすぎて維持できません。	• データベース管理者に連絡してください。 • ポーリング・プログラムをさらに追加します。
BatchQueue	スレッドを待機しているバッチの数が 0 より大きく、増加している。	スレッドの数が枯渇しています。これは、ダウンストリームの SNMP ディスパッチャーがキャパシティーの限界に近づいていることを示している可能性があります。	`NcPollerSchema.cfg` ファイルの **`BatchExtraThreads`** プロパティーを設定することで、スレッドの数を増やすことができますが、最適な解決方法ではありません。スレッドの数を増やすと、問題を悪化させる可能性があります。安全な解決方法は、以下のとおりです。 • ポーリング・プログラムをさらに追加します。 • システム管理者に連絡して、ホストへの RAM の追加を調査してもらいます。 **ヒント:** 処理のためにキューに入れられるバッチの数にしきい値を設定します。しきい値違反が発生すると、ポーリング・プログラム・ログにアラートが出力されます。[1]
PollDataQueueSize	キュー内の INSERT ステートメントの数が指数関数的に増大する。	データベースへの接続が失われたか、INSERT ステートメントの頻度がポーリング・プログラムの処理能力を超えています。	• データベース管理者に連絡してください。 • ポーリング・プログラムをさらに追加します。

表 8. ポーリング・プログラムのメトリックでの問題の考えられる原因 (続き)

メトリック	問題	考えられる原因	アクション
PollDataRowCount	プルーニングの完了後に、行数がしきい値を超えている。デフォルトのしきい値は 5,000,000 であり、デフォルトのプルーニング間隔は 1 時間です。	ポーリング・ロードが大きすぎるため、行数が多すぎてプルーニング間隔内に除去できません。あるいは、データベースで問題が発生したため、プルーニングの問題が発生しています。	データベース管理者に連絡してください。

表の注:

1. しきい値を設定するには、$NCHOME/etc/precision/NcPollerSchema.cfg 内の set **BatchQueueThreshold** プロパティーを適切な値に変更します。例えば、キューに入れられるポーリング・バッチのしきい値を 10 に設定するには、以下のようにします。

   ```
   update config.properties set BatchQueueThreshold = 10;
   ```

   指定されたしきい値をキューが超過した場合は、以下の例のようなメッセージが書き込まれます。

   ```
   2013-04-19T12:37:58:Poller:NCOMS:DataQueueSize:10;
   ```

エラーが表示された場合、$NCHOME/etc/precision/NcPollerSchema.cfg ファイルで、**CollectPollerMetrics** パラメーターが無効になっているかどうかを確認します。デフォルトではこのパラメーターは有効になっていますが、無効になっている場合は有効にしてください。OQL インターフェースを使用して、実行時にこのパラメーターを有効にすることができます。以下に例を示します。

```
ncp_oql -domain NCOMS -service SnmpPoller -poller Poller23507
-query "update config.properties set CollectPollerMetrics=1;"
```

関連タスク:

81 ページの『追加のポーリング・プログラムの設定』
ネットワークの負荷の処理に 1 つのポーリング・プログラムだけでは不十分な場合は、Network Manager サーバーで追加のポーリング・プログラムを設定します。ポーリング・プログラムを追加するには、**ncp_poller** プロセスの新しいインスタンスを登録して作成します。

37 ページの『ポーリング・ポリシーの変更』
ポーリング・ポリシー・エディターを使用して、既存のポーリング・ポリシーの設定を変更します。

75 ページの『ポーリング・データ・キューのサイズの調整』
NCPOLLDATA データベースに書き込むためのポーリング・データのキューが大きくなりすぎると、データベースの問題やポーリング・ロードが大きいことが原因で、ポーリング・プログラムがメモリー不足になることがあります。この問題をモニターして回避するために、キューのサイズについて制限を設定できます。この制限は、バッチの数として定義されます。制限を超過した場合、ログおよびアクティブ・イベント・リスト (AEL) でアラートが表示されます。その場合、ポーリング・プログラムは、キューに入れられたデータを破棄して、キューのサイズを減らします。破棄されたデータは、別のファイルに書き込まれるため、そのデータを取得してデータベースに書き込むことができます。

エンティティーの状況の照会

ポーリング・プログラムのトラブルシューティングを行うには、monitors.current OQL テーブルを照会します。このテーブルには、ポーリング・ポリシーのスコープに含まれるエンティティーのレコードが入っています。このテーブルを照会するには、**ncp_oql** コマンドを使用します。レコードには、エンティティーが最後にポーリングされた時刻などの情報が入っています。ポーリングされていないエンティティーの場合、出力には管理状況が表示されるか、エンティティーの IP アドレスが重複していると考えられると表示されます。

重要: Fix Pack 4 デフォルトでは、monitors.current テーブルに対する照会からシャーシ・レコードが出力されますが、モニター対象エンティティーのインターフェース・レコードは省略されます。シャーシ・レコードとインターフェース・レコードを両方とも出力するには、フィックスパック 4 を適用してください。

- monitors.current テーブルを照会するには、以下の例に示すように MASTER ドメインに対して **ncp_oql** コマンドを実行します。

```
ncp_oql -domain MASTER -service SnmpPoller -tabular
-query "select * from monitors.current;"
```

以下の例に示すように、シャーシ・レコードが出力されます。

```
| ENTITYID | ENTITYNAME | ACCESSIPADDRESS | MAINNODEENTITYID | MAINNODEENTITYNAME

| MAINNODEADDRESS | ENTITYMANAGED | CHASSISMANAGED | ISMANAGED | LASTPOLLTIME

| LASTPOLLINTERVAL | LASTPOLLFAILURE | TEMPLATEID | TEMPLATENAME | POLICYID

| POLICYNAME | CURRENTTIME |
| 27 | 172.30.120.15 | 172.30.120.15 | 27 | 172.30.120.15 | 172.30.120.15 |

| | true | 2013-10-23 16:38:24 | 180 | 2013-10-23 16:38:24 | 47 | aNoSuchName

| 179 | aNoSuchName | 2013-10-23 16:39:06 |

( 1 record(s) : Transaction complete )
```

Fix Pack 4 以下の例に示すように、最初にシャーシ・レコードが出力され、次にインターフェース・レコードが出力されます。

```
| ENTITYID | ENTITYNAME | ACCESSIPADDRESS | MAINNODEENTITYID | MAINNODEENTITYNAME

| MAINNODEADDRESS | ENTITYMANAGED | CH ASSISMANAGED | ISMANAGED | LASTPOLLTIME

| LASTPOLLINTERVAL | LASTPOLLFAILURE | TEMPLATEID | TEMPLATENAME | POLICYID
```

```
| POLICYNAME | CURRENTTIME | ENTITYCLASS | LINGERTIME | ENTITYTYPE | IFINDEX
| IFTYPESTRING | IFNAME | IFDESCR | IFALIAS  | INSTANCESTR | MONITOREDINSTID
|
| 27 | 172.30.120.15
| 172.30.120.15 | 27 | 172.30.120.15 | 172.30.120.15 | |   | true
| 2013-10-23 16:38:24 | 180 | 2013-10-23 16:38:24 | 47 | aNoSuchName | 179
| a NoSuchName | 2013-10-23 16:40:47 | NULL | NULL | NULL | NULL | NULL | NULL
| NULL | NULL   | NULL | NULL | | 1714 | 172.30.120.15[ 19 ] | | 27
| 172.30.120.15 | 172.30.120.15 | |   | true | NULL | NULL | NULL | 47
| aNoSuchName | 179 | a NoSuchName | NULL | CiscoCat19xx | 3 | 2 | 19
| ethernet-csmacd | 19 | 19 |   | 19 | 386 | | 1726 | 172.30.120.15[ CPU ]
| 172.30.120.15 | 27 | 172.30.120.15 | 172.30.120.15 | |   | true | NULL
| NULL | NULL | 47 | aNoSuchName | 179 | a NoSuchName | NULL | CiscoCat19xx
| 3 | 2 | 37 | ethernet-csmacd | CPU | CPU |   | 37 | 1536 |
( 3 record(s) : Transaction complete )
```

- Fix Pack 4 テーブルでシャーシ・レコードのみを照会するには、以下の例に示すコマンドを実行します。

```
ncp_oql -domain MASTER -service SnmpPoller -tabular
-query "select * from monitors.current where ENTITYTYPE=1;
```

第 10 章 ping ポーリングのトラブルシューティング

この情報を使用すると、ネットワーク内の重要な IP アドレスが Network Manager によって期待どおりに ping ポーリングされているかどうかを確認できます。また、期待どおりにポーリングされていない場合は、問題を解決するための情報を得ることができます。

デフォルトでは、`$NCHOME/precision/scripts/SQL/createPollLogTables.sql` ファイルに定義されている表とビューが、Network Manager のインストールの一環として NCMONITOR スキーマに追加されます。これらの表とビューには、この手順で実行する診断操作の結果が保管されます。

注: このメカニズムは、ping ポーリング・ポリシーにのみ適用されます。SNMP ポーリング・ポリシーには適用されません。

この手順には、指定したネットワーク・ドメイン内の現在の ping ポーリング状況のスナップショットを作成するステップが含まれています。Network Manager のポーリング・プロセスを開始したら、少なくとも 2 回のポーリング間隔の後で、最初のスナップショットを作成する必要があります。システムの再始動は不要です。

一連の IP アドレスを指定すると、このタスクで説明するスクリプトによって、ポーリング・プログラムがそれらの IP アドレスを ping しているかどうかを確認できます。 ping していない場合は、何が問題であるのかを知ることができます。IP アドレスがポーリングされない理由は多数ありますが、そのいくつかを以下に示します。

- ポーリング・ポリシーで指定されたデバイスがディスカバーされなかった。
- デバイスまたはインターフェースが ping ポリシー・スコープのいずれにも含まれていない。
- デバイスまたはインターフェースがトポロジー視覚化 GUI のいずれかから非管理としてマークされている。
- インターフェースがディスカバリー時に非管理としてマークされたか、またはルーティング不能と判別された。

このタスクで説明するスクリプトについて詳しくは、*IBM Tivoli Network Manager IP Edition* 管理ガイドを参照してください。

注: これらのスクリプトは、検証および診断ツールにすぎず、デバイスの実際のポーリングには影響しません。デバイスのポーリングは、ネットワーク・ポーリング GUI を使用してセットアップしたポーリング・ポリシーのみによって管理されます。

1. `ncp_upload_expected_ips.pl` スクリプトを使用してポーリングをモニターする IP アドレスのリストを追加します。 次のコマンドを発行します。 `$NCHOME/precision/bin/ncp_perl $NCHOME/precision/scripts/perl/scripts/ncp_upload_expected_ips.pl -domain DOMAIN_NAME -file FILENAME -password PASSWORD` それぞれの意味は、以下のとおりです。

© Copyright IBM Corp. 2006, 2016

- *DOMAIN_NAME* は、IP アドレスを含むドメインの名前です。 IP アドレスのリストは、このドメイン内の IP アドレスとのみ比較されます。
- *FILENAME* は、IP アドレスを含むプレーン・テキスト・ファイルです。各 IP アドレスは、空白文字で区切られています (例えば、1 行ごとに 1 つの IP アドレス)。 IPv4 アドレスのみ使用できます。このファイルには、標準ドット表記の IP アドレスのみ含まれている必要があります。
- *PASSWORD* は、NCIM および NCMONITOR スキーマにアクセスする際に使用されるデータベース・パスワードです。

注**:** この操作は、ポーリングをモニターする IP アドレスのリストが変更された場合にのみ繰り返してください。それ以外の場合、この操作は、ドメインごとに 1 回のみ行ってください。

2. 前のステップで指定したドメイン内の現在の ping ポーリング状況のスナップショットを、定期的にまたはトラブルシューティングのために必要になったときに作成します。 次のコマンドを発行します。`$NCHOME/precision/bin/ncp_perl $NCHOME/precision/scripts/perl/scripts/ncp_ping_poller_snapshot.pl -domain` *DOMAIN_NAME* `-password` *PASSWORD* それぞれの意味は、以下のとおりです。
   - *DOMAIN_NAME* は、現在の ping ポーリング状況のスナップショットを作成するドメインの名前です。
   - *PASSWORD* は、NCIM および NCMONITOR スキーマにアクセスする際に使用されるデータベース・パスワードです。

   操作の結果は、NCMONITOR スキーマ内の pollLog データベース表に保管されます。
3. ポーリングされないエンティティーについてのレポート。 Network Manager のポーリング・プロセスでポーリングされたエンティティーやポーリングされなかったエンティティーの状況を示すレポートを実行します。 次のコマンドを発行します。`$NCHOME/precision/bin/ncp_perl $NCHOME/precision/scripts/perl/scripts/ ncp_polling_exceptions.pl -domain` *DOMAIN_NAME* `[ -notpolled ] [-format LIST | REPORT ]` それぞれの意味は、以下のとおりです。
   - *DOMAIN_NAME* は、現在の ping ポーリング状況をレポートするドメインの名前です。
   - `-notpolled`: このオプション・パラメーターは、予期される IP アドレスのリストと比較して、ポーリングされない IP アドレスのリストを出力します。これは、リスト形式でのみ出力されます。
   - `-format LIST | REPORT`: このオプション・パラメーターは、出力をレポート形式とリスト形式のいずれにするかを指定します。

   このコマンドは 2 つのリスト、つまり、ポーリングする IP アドレスのリストと、ポーリングされない IP アドレスのリストを出力します。ポーリング対象の IP アドレスのうちでポーリングされていない IP アドレスがないか、一目で確認できます。

関連資料:

252 ページの『ping ポーリング状況テーブル』

NCMONITOR ping ポーリング状況テーブルを使用して、ネットワーク ping ポーリングで診断操作を実行することができます。

第 11 章 イベント・エンリッチおよびイベント相関について

Network Manager はイベント・ゲートウェイを使用して、イベントをトポロジー・データと相関させ、デフォルトのトポロジー・フィールド・セットによってイベントを強化します。強化されたイベントは、根本原因分析 (RCA) やフェイルオーバーなどのプラグイン・プロセスに渡され、強化されたイベントに含まれているデータに基づいてさらなるアクションが行われます。強化されたイベントも ObjectServer に戻されます。

イベントの強化

イベント・エンリッチ・プロセスはイベント・ゲートウェイ内で実施され、別個のステップで構成されます。 これらの各ステップはカスタマイズが可能です。

関連概念:

241 ページの『付録 F. Network Manager のイベント・カテゴリー』
Network Manager によって生成されるイベントは、2 つのカテゴリーに分類されます。1 つはモニターされているネットワークに関するイベント、もう 1 つは Network Manager のプロセスに関するイベントです。

イベント・エンリッチのクイック・リファレンス

ここでは、イベントがイベント・ゲートウェイを通過するときにどのように処理されるかを説明します。

注: ファイアウォールで保護されている Tivoli Netcool/OMNIbus ObjectServer にアクセスする場合、IDUC ポートを指定し、ファイアウォールを使用して、そのポートへのアクセスを提供する必要があります。ObjectServer の IDUC ポートの指定について詳しくは、「*IBM Tivoli Netcool/OMNIbus* 管理ガイド」を参照してください。

以下の表に、手順の説明を示します。

© Copyright IBM Corp. 2006, 2016

表 9. イベント・エンリッチのクイック・リファレンス

アクション	その他の情報	次のステップに渡されるデータ
1. ObjectServer サーバーからイベントを受け取り、そのイベントに着信イベント・フィルターが適用されます。 デフォルトのフィルターがイベントの LocalNodeAlias フィールドを調べます。 LocalNodeAlias フィールドが空でない場合、イベントはフィルターを通過し、ステップ 3 に進みます。 **注:** 通常 LocalNodeAlias フィールドには、イベントが発生したメイン・ノード・デバイスを指すデータが含まれています。 LocalNodeAlias フィールドに保持されている精密なデータはさまざまであり、以下が含まれている可能性があります。 • IP アドレス • DNS 名 • sysName	102 ページの『着信イベント・フィルター』	イベント
2. イベント・ゲートウェイは、イベント内の重大度フィールドおよび合計フィールドに基づいてイベントに状態を割り当てます。 このイベント状態はイベント・ゲートウェイの内部表現であり、イベント・サブスクリプション・メカニズムの一環として後でプラグインによって使用されます。	110 ページの『イベント状態』	イベント イベント状態
3. イベントに着信フィールド・フィルターが適用されます。 このフィールド・フィルターは、イベント・ゲートウェイの処理に関与していない alerts.status フィールドをフィルターで除外します。	106 ページの『着信フィールド・フィルター』	フィルター処理されたフィールドを持つイベント イベント状態
4. イベント・ゲートウェイは使用するイベント・マップを決定して、このイベントの処理方法を決定します。 イベント・マップはイベントの処理方法を定義します。 同時に、イベントに数値の優先順位値が関連付けられます。 この優先順位値は、同じエンティティーに対して複数のイベントがある場合に RCA プラグインによって使用されます。 エンティティーで最高の優先順位値を持つイベントは、他のイベントの抑止に使用されます。	114 ページの『イベント・マップ選択』 156 ページの『優先順位値』	フィルター処理されたフィールドを持つイベント イベント状態 イベント・マップ・フィールド (イベント・マップ名やイベント・エンリッチ・スティッチャーなど) 優先順位値

表 9. イベント・エンリッチのクイック・リファレンス (続き)

アクション	その他の情報	次のステップに渡されるデータ
5. イベント・ゲートウェイは、Network Manager サーバーまたは入口インターフェースのエンティティー ID を決定します。入口インターフェースは、Network Manager サーバーとの間でネットワーク・パケットの送受信が行われるディスカバリー・スコープ内のインターフェースです。 この値は RCA プラグインによって、分離抑止を実行するために使用されます。	158 ページの『ポーリング・プログラム・エンティティー』	フィルター処理されたフィールドを持つイベント イベント状態 イベント・マップ・フィールド (イベント・マップ名やイベント・エンリッチ・スティッチャーなど) 優先順位値 PollerEntityId
6. イベント・ゲートウェイはトポロジー・ルックアップを実行してこのイベントに関連付けられているエンティティー・データを取得した後、このエンティティー・データの一部を使用してイベントを強化します。 イベント・ゲートウェイはトポロジー・ルックアップとイベント・エンリッチを実行するために、イベント・マップに定義されているスティッチャーを呼び出します。	121 ページの『イベント・ゲートウェイ・スティッチャー』	ステップ 8 およびステップ 9 へ フィルター処理されたフィールドおよび強化されたフィールドを持つイベント イベント状態 優先順位値 PollerEntityId スティッチャーからの戻り値
7. イベントに発信フィールド・フィルターが適用されます。 このフィルターは、イベント・ゲートウェイ (特に GwEnrichEvent スティッチャー・ルール) によって強化されたフィールドのみを渡します。 強化されたフィールドはイベント・ゲートウェイ・キューに入れられ、そこから、一定の間隔 (構成可能であり、デフォルトは 5 秒) でデータが ObjectServerに送信されます。	107 ページの『発信フィールド・フィルター』 109 ページの『出力イベント・ゲートウェイ・キュー』	イベント・ゲートウェイによって強化されたフィールド
8. イベント・マップに定義されたスティッチャーからの戻り値に基づいて (ステップ 7)、イベント・ゲートウェイは強化したイベントをプラグインに送信するかどうかを決定します。 • 戻り値が 1 の場合、イベント・ゲートウェイは強化したイベントをプラグインに送信します。 次の手順に進みます。 • 戻り値が 0 の場合、イベント・ゲートウェイは強化したイベントをプラグインに送信しません。 このイベントのイベント・エンリッチ・プロセスはここで終了します。	133 ページの『イベント強化スティッチャー』	フィルター処理されたフィールドおよび強化されたフィールドを持つイベント イベント状態 イベント・マップ名 優先順位値 PollerEntityId

表 9. イベント・エンリッチのクイック・リファレンス (続き)

アクション	その他の情報	次のステップに渡されるデータ
9. 各プラグインが、強化されたイベントを受け取りたいかどうかを決定します。 この判断は、イベント・マップ名とイベント状態に基づいて行われます。 イベントを受け取ったプラグインは、さらなるイベント・エンリッチまたはその他のアクションを実行します。	140 ページの『プラグインの説明』 152 ページの『プラグインのサブスクリプション』	フィルター処理されたフィールドおよび強化されたフィールドを持つイベント
10. 処理が完了すると、強化されたフィールドはイベント・ゲートウェイ・キューに入れられ、そこから、一定の間隔 (構成可能であり、デフォルトは 5 秒) でデータが ObjectServer に送信されます。	109 ページの『出力イベント・ゲートウェイ・キュー』	ObjectServer へ プラグインによって強化されたフィールド

関連タスク:

193 ページの『イベント・マップ・サブスクリプションの変更』
プラグインがサブスクライブしているイベント・マップを変更することができます。例えば、新しいイベント・マップを追加し、そのイベント・マップによって処理されるイベントに対してシステムによる RCA を実行する場合は、そのイベント・マップを RCA プラグインのサブスクリプション・リストに追加する必要があります。

イベント・フィルター

イベントが ObjectServer から受け取られると、強化プロセスの最初にフィルター処理されます。また、強化されたイベントが ObjectServer に送り返されると、そのプロセスの最後にイベントがフィルター処理されます。

着信フィルター

着信フィルターは、イベント・エンリッチのために必要なイベントおよびフィールドのみがイベント・ゲートウェイに渡されるようにします。

着信イベント・フィルター**:**

着信イベント・フィルターは、ObjectServer からのイベントを除去し、特定の基準を満たすイベントのみを通過させます。

着信イベント・フィルターは、フェイルオーバーを行わないインストール済み環境とフェイルオーバーを行うインストール済み環境の両方で使用されます。 フェイルオーバーを行うインストール済み環境の場合、フィルター・メカニズムはアクティブ・ドメイン (つまり、プライマリー・サーバーが正常な場合はプライマリー・ドメイン、またはプライマリー・サーバーがダウンしている場合はバックアップ・ドメイン) 上のイベントに適用されます。

着信イベント・フィルターは、以下の条件を満たすイベントのみを通過させます。

- イベントの LocalNodeAlias フィールドに値が取り込まれている。 このフィールドには、関連付けられたデバイスの IP アドレスまたは DNS 名が保持されているはずです。
- イベントの NmosDomainName フィールドに指定されているドメイン名が、現在のイベント・ゲートウェイによって処理されているドメインと同じである。

あるいは、このイベントにドメインが関連付けられておらず、NmosDomainName フィールドが空である。

着信イベント・フィルターは、単一ドメイン・システムと複数ドメイン・システムの両方のイベントを処理できます。

単一ドメインおよび複数ドメインでの処理

単一ドメイン・システムの場合、イベント・ゲートウェイ・プロセスは 1 つのみです。 NmosDomainName フィールドに設定されているドメインを含むイベントは、すべてこのイベント・ゲートウェイと同じドメイン割り当てを持っています。

複数ドメイン・システムでは、ドメインごとに 1 つずつ、複数のイベント・ゲートウェイ・プロセスがあります。 各イベント・ゲートウェイ・プロセスは ObjectServer からイベントを受け取ってフィルター処理し、それ自身のドメインのイベントのみを受け取るようにします。 ObjectServer は、alerts.status フィールド AlertKey、Identifier、および Domain に基づいて、問題イベントと解決イベントの非重複化および突き合わせを実行します。 Domain フィールドを含めることにより、問題イベントと解決イベントのすべての非重複化および突き合わせがドメイン固有に行われます。

ドメインを含まないイベント

Network Manager の外部のソースから送られるイベント (例えば、Tivoli Netcool/OMNIbus syslog やトラップ・プローブなど) には、それらが初めてイベント・ゲートウェイを通過する時点ではドメイン設定が含まれていません。 単一ドメイン・システムでは、ドメイン設定を含まないイベントは着信イベント・フィルターを通過し、イベント・エンリッチ中に実行されるトポロジー・ルックアップにおいて、イベント・ゲートウェイがイベントと関連付けるドメインを決定します。

複数ドメイン・システムでは、ドメインを含まないイベントを最初に検出したイベント・ゲートウェイが、そのイベントを着信イベント・フィルターを通過させ、続いて処理します。 ただし、トポロジー・ルックアップでイベントに対するエンティティーが見つからなかった場合、イベント・ゲートウェイはイベント・エンリッチを行わずにそのイベントを拒否します。 その後、別のイベント・ゲートウェイがそのイベントを受け取り、同様に、そのイベントをそれ自身のドメイン内のデバイスと突き合わせしようとします。このプロセスは、そのイベントとエンティティーとの突き合わせを行うことができるイベント・ゲートウェイによって、最終的にそのイベントが処理されるまで続けられます。

関連資料:

273 ページの『config.nco2ncp テーブル』
config.nco2ncp テーブルは、Tivoli Netcool/OMNIbus からNetwork Manager に渡されるイベントをフィルタリングするために使用されます。

着信イベント・フィルター**:** デフォルト構成**:**

この例は、着信イベント・フィルターを EventGatewaySchema.cfg 構成ファイルで構成する方法を示しています。着信イベント・フィルターのこの標準的な insert は、単一ドメイン・システムと複数ドメイン・システムの両方を処理します。

着信イベント・フィルターは EventGatewaySchema.cfg 構成ファイルで構成されます。 このファイルの場所は、`$NCHOME/etc/precision/EventGatewaySchema.cfg`です。

次の表では、この insert の関連する行について説明します。

表 *10.* 着信イベント・フィルターに関連するコードの行

行番号	説明
1	config.nco2ncp テーブルへの insert を作成して、着信フィルターを構成します。
3	config.nco2ncp テーブルの EventFilter フィールド内への insert を指定します。
9 - 10	以下のようにフィルターを設定します。 `"LocalNodeAlias <> '' and (NmosDomainName = '$DOMAIN_NAME' or NmosDomainName = '')"` この節では、イベントの LocalNodeAlias 列に値が取り込まれていることが検査されます。さらに、イベントに指定されているドメイン名 (NmosDomainName フィールドに保持) が、イベント・ゲートウェイによって処理されているドメイン (`$DOMAIN_NAME` 変数に保持) と同じであるかどうかが、このフィルターによって検査されます。一致する場合、またはイベントに関連付けられたドメインがない (`NmosDomainName - ''`) 場合に、LocalNodeAlias 列に値が取り込まれていると、イベントはフィルターを通過します。

次の表では、この insert の関連する行について説明します。

```
1]    insert into config.nco2ncp
2]    (
3]        EventFilter,
4]        StandbyEventFilter,
5]        FieldFilter
6]    )
7]    values
8]    (
9]        "LocalNodeAlias <> '' and (NmosDomainName =
10]            '$DOMAIN_NAME' or NmosDomainName = '')",
11]       "EventId in ('ItnmHealthChk', 'ItnmDatabaseConnection')",
12]       [
13]           "Acknowledged",
14]           "AlertGroup",
15]           "EventId",
16]           "FirstOccurrence",
17]           "LastOccurrence",
18]           "LocalNodeAlias",
19]           "LocalPriObj",
20]           "LocalRootObj",
21]           "Manager",
22]           "NmosCauseType",
23]           "NmosDomainName",
24]           "NmosEntityId",
25]           "NmosEventMap",
26]           "NmosManagedStatus",
27]           "NmosObjInst",
28]           "NmosSerial",
29]           "Node",
30]           "RemoteNodeAlias",
31]           "EventId",
32]           "Serial",
33]           "ServerName",
34]           "Severity",
35]           "Summary",
36]           "SuppressEscl",
37]           "Tally",
38]           "Type"
39]       ]
40]   );
```

スタンバイ・フィルター:

フェイルオーバーのデプロイメントでは、スタンバイ・フィルターがフェイルオーバー・ペアのバックアップ・ドメインで使用されます。これは、プライマリーがアクティブである場合はバックアップ・ドメイン、バックアップがアクティブである場合はプライマリー・ドメインを意味します。スタンバイ・フィルターでは、正常性検査 (ItnmHealthCheck) イベントのみが Event Gateway を通過することができます。これらのイベントはフェイルオーバー・プラグインに渡され、プライマリー・モードにスイッチバックするようシステムに指示します。フェイルオーバーの動作のために、スタンバイ・フィルターに変更を行う場合は、このフィルターが引き続き正常性検査イベントを受け入れるようにする必要があることに注意してください。

プライマリー・サーバーがアクティブな場合、バックアップ・サーバーのイベント・ゲートウェイはイベント・エンリッチを実行しません。プライマリー・サーバーがダウンし、バックアップ・サーバーがアクティブな場合、バックアップ・サーバーのイベント・ゲートウェイは ObjectServer でイベント・エンリッチを実行します。

スタンバイ・フィルターは EventGatewaySchema.cfg 構成ファイルで構成されます。このファイルの場所は、$NCHOME/etc/precision/EventGatewaySchema.cfgです。

104 ページの『着信イベント・フィルター: デフォルト構成』に示されている例は、スタンバイ・フィルターを EventGatewaySchema.cfg 構成ファイルで構成する方法を示しています。

スタンバイ・フィルターに関連するコード・セクションを、以下の行にリストします。この insert は、プライマリー・サーバーがダウンし、バックアップ・サーバーがアクティブな場合に ItnmHealthCheck イベントのみを通過させるようにイベント・ゲートウェイを構成します。

次の表では、この insert の関連する行について説明します。

表 *11.* スタンバイ・フィルターに関連するコードの行

行番号	説明
1	config.nco2ncp テーブルへの insert を作成して、着信フィルターを構成します。
4	config.nco2ncp テーブルの StandbyEventFilter フィールド内への insert を指定します。
11	以下のようにフィルターを設定します。 `"EventId in ('ItnmHealthChk', 'ItnmDatabaseConnection')",` この節により、イベント ID が値 ItnmHealthChk または ItnmDatabaseConnection に設定されたイベントのみが通過します。

関連資料:

273 ページの『config.nco2ncp テーブル』
config.nco2ncp テーブルは、Tivoli Netcool/OMNIbus からNetwork Manager に渡されるイベントをフィルタリングするために使用されます。

着信フィールド・フィルター**:**

着信イベント・フィルターを通過するイベントごとに、このフィールド・フィルターは、通過させてイベント・エンリッチ・プロセスに渡す alerts.status フィールドのサブセットを指定します。 このフィールド・フィルターが空の場合、すべての alerts.status フィールドが通過してイベント・エンリッチ・プロセスに渡されます。このフィルターの目的は、渡されるフィールドを必要最小限に抑えて、処理の負荷を軽減することです。

着信イベント・フィルターは EventGatewaySchema.cfg 構成ファイルで構成されます。 このファイルの場所は、$NCHOME/etc/precision/EventGatewaySchema.cfgです。

「 104 ページの『着信イベント・フィルター: デフォルト構成』」に示されている例は、着信フィールド・フィルターを EventGatewaySchema.cfg 構成ファイルで構成する方法を示しています。

着信フィールド・フィルターに関連するコード・セクションを、この例の以下の行にリストします。 ObjectServer がイベント・ゲートウェイに渡すイベントごとに、着信フィールド・フィルターは、通過させてイベント・エンリッチ・プロセスに渡す alerts.status フィールドのサブセットを指定します。

次の表では、この例の関連する行について説明します。

表 12. 着信フィールド・フィルターに関連するコードの行

行番号	説明
1	config.ncp2nco テーブルへの insert を作成して、着信フィルターを構成します。
5	config.ncp2nco テーブルの FieldFilter フィールド内への insert を指定します。
11-38	行 13 から 38 に指定されている alerts.status フィールドのみを通過させます。 **注:** イベント・エンリッチを追加で構成する場合、このリストにフィールドを追加する必要が生じる可能性があります。

発信フィールド・フィルター

発信フィールド・フィルターは、イベント・ゲートウェイによって更新される可能性のある ObjectServer フィールド・セットを定義します。

イベント・エンリッチは `GwEnrichEvent()` スティッチャー・ルールによって実行されます。 このルールによって強化されるフィールドのうち、このフィルターにリストされているフィールドのみが Object Server に送信されます。 一部のイベント・エンリッチは、イベント・ゲートウェイ・プラグインが使用するデータを提供することのみを目的としています。プラグインが使用することのみを目的として強化されたフィールドは、発信フィールド・フィルターに指定する必要はありません。 発信フィールド・フィルターは `GwEnrichEvent()` ルールに渡されるデータに対して作用します。

注**:** 強化されたフィールドをイベントに追加するためにイベント・エンリッチをカスタマイズした場合、これらの強化された追加のフィールドを含めるように発信フィールド・フィルターを更新する必要があります。 例えば、Cisco ルーターに関するアラートをすべて「重大」重大度 (重大度 5) に格上げするようにイベントを強化する場合、発信フィールド・フィルターのフィールドのリストに「重大度」を追加する必要があります。

発信フィールド・フィルターは EventGatewaySchema.cfg 構成ファイルで構成されます。 このファイルの場所は、`$NCHOME/etc/precision/EventGatewaySchema.cfg`です。

次の例は、発信フィールド・フィルターを EventGatewaySchema.cfg 構成ファイルで構成する方法を示しています。

発信フィールド・フィルターに関連するコード・セクションを、この例の以下の行にリストします。 強化されたフィールドがイベント・ゲートウェイによって ObjectServer に戻されるたびに、このフィルターにリストされているフィールドのみが送信されます。 その他のフィールドは無視されます。

次の表では、この例の関連する行について説明します。

表 *13.* 着信イベント・フィルターに関連するコードの行

行番号	説明
1	config.ncp2nco テーブルへの insert を作成して、発信フィールド・フィルターを構成します。
3	config.ncp2nco テーブルの FieldFilter フィールド内への insert を指定します。
5-14	行 5 から 14 に指定されているフィールドのみを ObjectServerに戻します。 **注:** イベント・エンリッチを追加で構成する場合、このリストにフィールドを追加する必要があります。

```
1]    insert into config.ncp2nco
2]    (
3]        FieldFilter
4]    )
5]    values
6]    (
7]        [
8]            "NmosCauseType",
9]            "NmosDomainName",
10]           "NmosEntityId",
11]           "NmosManagedStatus",
12]           "NmosObjInst",
13]           "NmosSerial"
14]       ]
15]   );
```

関連タスク:

182 ページの『例: メイン・ノード・デバイスのロケーションによるイベントの強化』
イベントに関連付けられているメイン・ノード・デバイスのロケーションがイベントのフィールドに追加されるように、イベント・エンリッチを構成することができます。

184 ページの『例: インターフェース名によるイベントの強化』
すべてのインターフェース・イベントに関してイベントが発生したインターフェースの名前がそのイベントのフィールドに追加されるように、イベント・エンリッチを構成することができます。

関連資料:

275 ページの『config.ncp2nco テーブル』
config.ncp2nco テーブルは、Network Manager IP Edition からTivoli Netcool/OMNIbus に渡されるイベントをフィルタリングし、マップするために使用されます。

出力イベント・ゲートウェイ・キュー:

出力イベント・ゲートウェイ・キューは、イベント・ゲートウェイ・スティッチャー (主となるイベント・エンリッチ) およびプラグインから、強化されたイベントを受け取ります。 更新の数を最小限にして ObjectServer への負荷を最小限に抑えるために、Object Server への更新はキューに入れられ、集約されたうえで、指定した間隔で ObjectServer に送信されます。 デフォルトは 5 秒です。

次の図に示すように、出力イベント・ゲートウェイ・キューは、イベント・ゲートウェイ・スティッチャー (主となるイベント・エンリッチ) とプラグインの両方から、強化されたイベントを受け取ります。

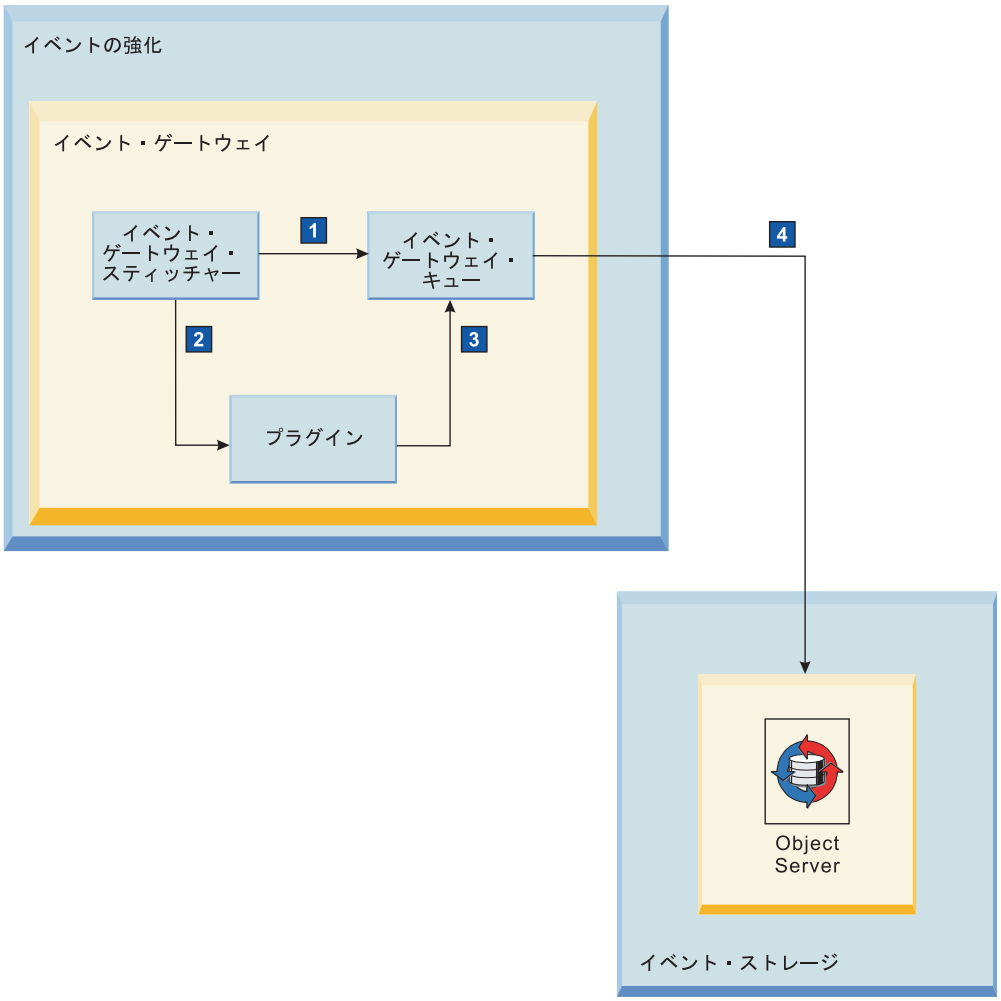


図 2. 出力イベント・ゲートウェイ・キュー

1 標準の強化されたイベントがイベント・ゲートウェイ・キューに入れられる
標準のイベント・エンリッチの後、イベント・ゲートウェイ・スティッチャーは強化されたデータをイベント・ゲートウェイ・キューに入れます。

2 イベントがプラグインに渡される
イベント・ゲートウェイ・スティッチャーが値 1 を返した場合、関連付け

られたイベントはプラグインに渡され、さらなる強化が行われます。 プラグインは、関連付けられたイベント・マップとイベント状態に基づいて、強化するイベントを選択します。

3 プラグインによって強化されたイベントがイベント・ゲートウェイ・キューに入れられる

プラグインは、強化されたイベントをイベント・ゲートウェイ・キューに入れます。

4 強化されたイベントが **ObjectServer** に送信される

強化されたイベントのデータは 5 秒ごとに ObjectServer に送信されます。5 秒という間隔は構成可能です。

デフォルトでは、イベント・ゲートウェイ・スティッチャーが NmosManagedStatus、NmosEntityId、NmosObjInst、および NmosDomainName フィールドを取り込んでイベントを強化します。 これらのイベントはイベント・ゲートウェイ・キューに入れられ、次の更新まで待機します。 その間に、イベントは RCA プラグインに渡されます。 RCA プラグインは、NmosSerial および NmosCauseType フィールドを取り込んでイベントを強化した後、それらのイベントをイベント・ゲートウェイ・キューに入れます。 これらの更新はいずれも、1 回の 5 秒間隔内でイベント・ゲートウェイ・キューに到着しています。 その結果、更新を 2 回実行する代わりに、イベント・ゲートウェイはデータをキューで待機させることができるため、これらのすべてのフィールドについて実行する ObjectServer の更新は 1 回だけで済みます。

関連タスク:

185 ページの『ObjectServer 更新間隔フィールドの構成』
イベント・ゲートウェイが ObjectServer に対するイベント・エンリッチの更新をキューに入れる際の間隔を構成することができます。

193 ページの『イベント・マップ・サブスクリプションの変更』
プラグインがサブスクライブしているイベント・マップを変更することができます。例えば、新しいイベント・マップを追加し、そのイベント・マップによって処理されるイベントに対してシステムによる RCA を実行する場合は、そのイベント・マップを RCA プラグインのサブスクリプション・リストに追加する必要があります。

イベント状態

イベント・ゲートウェイは、イベントのタイプ、およびイベント内の重大度フィールドと合計フィールドに基づいて、イベントに状態を割り当てます。 イベント状態は、イベント・プラグインによってイベントのサブスクライブ時に使用されるパラメーターの 1 つです。

関連概念:

164 ページの『RCA スティッチャーの説明』
ここでは、各 RCA スティッチャーの仕組みについて説明します。

162 ページの『RCA スティッチャーのシーケンス』
ここでは、RCA プラグインによって呼び出されるスティッチャーについて、および根本原因を判別するためにスティッチャーが実行されるシーケンスについて説明します。

イベント・タイプ

イベント・ゲートウェイのために、イベント・タイプは「問題」、「解決」、および「情報」という 3 つのおおまかなカテゴリーに分類されます。

イベント・ゲートウェイのために、イベント・タイプは以下のカテゴリーに分類されます。

タイプ **= 1:** 問題
: イベント・ゲートウェイは、以前の状態、およびイベント内の重大度フィールドと合計フィールドに基づいて、問題イベントにいくつかの状態のうち 1 つを割り当てます。

タイプ **= 2:** 解決
: 解決イベントには、直ちに「Resolution」状態が割り当てられます。

タイプ **=13:** 情報
: 情報イベントには、直ちに「情報」状態が割り当てられます。

alerts.status テーブルで定義されているすべてのイベント・タイプのリストを次の表に示します。各イベント・タイプは、上記のリストにある 3 つのカテゴリーのうち 1 つにマップされます。

表 14. 遷移ラベル

alerts.status でのタイプ・フィールドの値	イベント・ゲートウェイが認識するイベント・タイプ
0: タイプは設定されていません	不明
1: 問題	問題
2: 解決	Resolution
3: Netcool/Visionary の問題	問題
4: Netcool/Visionary の解決	Resolution
7: Netcool/ISM の新規アラーム	問題
8: Netcool/ISM の古いアラーム	問題
11: 重大度が高い	問題
12: 重大度が低い	問題
13: 情報	情報

イベント状態遷移図

イベント状態遷移図は、可能性のあるイベント状態を示し、alerts.status の重大度フィールドと合計フィールドの値に基づいて、これらの状態間での遷移がどのように行われるかについて説明します。この図は、さまざまなイベント・タイプの処理方法も示しています。

イベント状態遷移図を以下に示します。各イベントには、イベント・ゲートウェイを通過する間に、以下のいずれかの状態が割り当てられます。それぞれの状態遷移は、ObjectServer から受け取る更新済みイベントに対応しています。イベント状態には、以下の色が関連付けられています。

- 赤は、アクティブな問題状態を示します。
- 緑は、アクティブなクリア状態を示します。
- 白は、それがアクティブ状態でないことを示します。

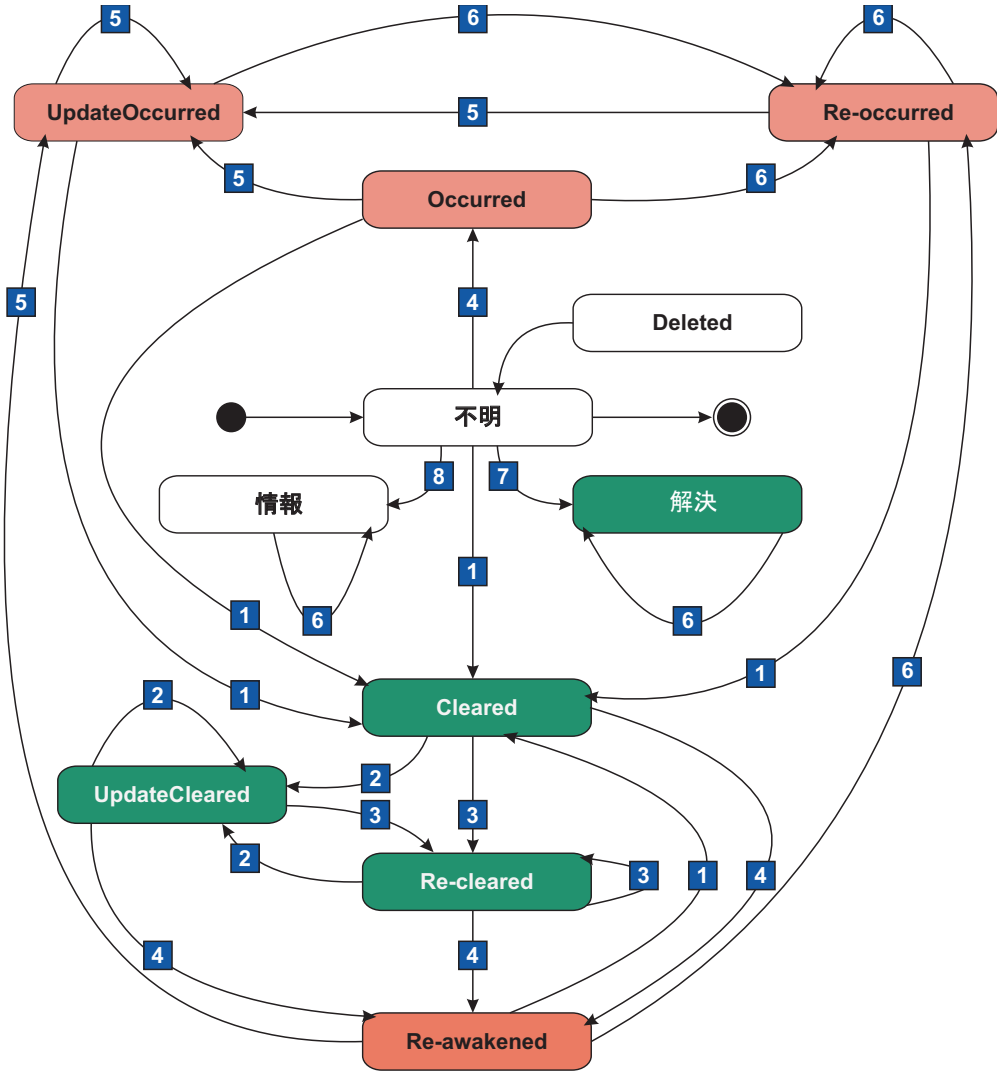


図 3. イベント状態遷移図

イベント状態遷移

この図では、それぞれの遷移に数字の 1 から 8 までのラベルが付いています。各ラベルに対応する遷移を次の表にリストします。

表 15. 遷移ラベル

ラベル	更新済みイベントのフィールド値
1	重大度 = 0
2	重大度 = 0 合計: 前のイベントから変更なし
3	重大度: 0 合計: 前のイベントから変更あり
4	重大度: 非ゼロ
5	重大度: 非ゼロ 合計: 前のイベントから変更なし
6	重大度: 非ゼロ 合計: 前のイベントから変更あり
7	タイプ = 2: 解決
8	タイプ =13: 情報

イベント状態の説明

イベント状態を以下のテーブルに示します。特に明記しない限り、リストされている状態はすべて、タイプ = 1 のイベント (問題イベント) を示しています。

表 16. イベント状態

状態	説明
Cleared	重大度ゼロのイベントが以前にイベント・ゲートウェイに認識されていなかったか、またはアクティブな問題状態でした。
Deleted	イベントが ObjectServer から削除されました。これは随時発生すると考えられるため、「Unknown」を除く他のどの状態からもこの状態に遷移する可能性があります。削除はプラグイン・インターフェースを介してプラグインに送信され、イベント・ゲートウェイ内のイベントの状態は「Unknown」になります。
情報	タイプが「情報」(タイプ = 13) である着信イベントには、他のフィールド値に関係なく、「Information」の状態が割り当てられます。
Occurred	重大度が非ゼロのイベントが以前にイベント・ゲートウェイに認識されていませんでした。
Re-awakened	重大度が非ゼロのイベントが以前にイベント・ゲートウェイに認識されていましたが、アクティブな問題状態ではありませんでした。
Re-cleared	重大度ゼロのイベントが以前にイベント・ゲートウェイに認識されていて、再び発生しました。

表 16. イベント状態 (続き)

状態	説明
Re-occurred	重大度が非ゼロのイベントが以前にイベント・ゲートウェイに認識されていて、その問題イベントが新たに発生しました。
Re-synchronize	イベント・ゲートウェイが ObjectServer と再同期しました。これは、単一のどの ObjectServer イベントにも対応しない合成状態です。
Resolution	タイプが「解決」(タイプ = 2) である着信イベントには、他のフィールド値に関係なく、「Resolution」の状態が割り当てられます。
不明	イベント・ゲートウェイによってイベントが検出されていません。これは初期かつ最終の状態です。
UpdateCleared	重大度ゼロのイベントが以前にイベント・ゲートウェイに認識されていて、再発生ではなく、更新が検出されました。
UpdateOccurred	重大度が非ゼロのイベントが以前にイベント・ゲートウェイに認識されていて、再発生ではなく、更新が検出されました。

イベント処理

イベント処理には、各イベント・タイプを ObjectServer からトポロジー・データ内のエンティティーにマップする方法を決定する処理が含まれます。

イベント・マップ

イベント処理はイベント・マップを使用して実行されます。イベント・マップの主な機能は、トポロジー・ルックアップを実行する 1 組のスティッチャーを呼び出して、イベントに関連付けるエンティティーを決定し、トポロジー・データを使用してイベントを強化することです。

イベント・マップ選択:

イベント・ゲートウェイは、alerts.status EventId フィールドに定義されているイベントの種類に基づいて、使用するイベント・マップを決定します。イベントの種類の一例として、SNMP トラップ・リンクダウン・イベントがあります。

イベント・ゲートウェイを使用するイベント・マップの選択:

この情報を使用して、イベント処理に使用するイベント・マップを選択するためにイベント・ゲートウェイを構成する方法を説明します。

イベント・ゲートウェイを使用してイベント・マップの選択を構成することにした場合、イベント・ゲートウェイの config.precedence テーブルを構成する必要があります。 config.precedence テーブルは EventGatewaySchema.cfg 構成ファイル内で構成します。このファイルの場所は、`$NCHOME/etc/precision/EventGatewaySchema.cfg`です。

次の例は、config.precedence テーブルを EventGatewaySchema.cfg 構成ファイル内で構成する方法を示しています。

イベント処理に使用するイベント・マップの選択に関連するコード・セクションを、この例の以下の行にリストします。この例の insert は、EventId フィールドが

SNMPTRAP-LinkDown に設定されているすべてのイベントに対して、イベント・マップ LinkDownIfIndex を使用するようにイベント・ゲートウェイを構成します。これらは Tivoli Netcool/OMNIbus プローブから発信されるトラップ・イベントです。

次の表では、この例の関連する行について説明します。

表 17. 着信イベント・フィルターに関連するコードの行

行番号	説明
1	config.precedence テーブルへの insert を作成して、着信フィルターを構成します。
4-5	config.precedence テーブルの EventMapName および NcoEventIds フィールド内への insert を指定します。
10	EventMapName フィールドを値 LinkDownIfIndex に設定します。
11	NcoEventId フィールドを alerts.status テーブル内の EventId フィールドの値に設定します。この例では、alerts.status の EventId 値が SNMPTRAP-LinkDown であるすべてのイベントに対して、イベント・マップ LinkDownIfIndex が選択されています。

```
1]    insert into config.precedence
2]    (
3]        Precedence,
4]        EventMapName,
5]        NcoEventId
6]    )
7]    values
8]    (
9]        910,
10]       "LinkDownIfIndex",
11]       "SNMPTRAP-LinkDown"
12]   );
```

イベント・マップの選択方法*:*

イベントを処理するためにイベント・マップを選択するには、eventPrecedence insert を指定します。eventPrecedence insert は、Tivoli Netcool/OMNIbus プローブ・ルール・ファイル、IBM Tivoli Netcool/OMNIbus ナレッジ・ライブラリー、またはイベント・ゲートウェイのどれを構成しても指定できます。

イベントを処理するためにイベント・マップを選択するには、eventPrecedence insert を指定します。eventPrecedence insert は、以下のいずれかを構成することによって指定できます。

Tivoli Netcool/OMNIbus プローブのルール・ファイル
: プローブ・ルール・ファイルを構成して、alerts.status イベント・レコードの NmosEventMap フィールドにイベント・マップの名前とオプションの優先順位値を取り込みます。これが eventPrecedence insert に相当します。

IBM Tivoli Netcool/OMNIbus ナレッジ・ライブラリー
: Netcool/OMNIbus ナレッジ・ライブラリー 構成ファイル NcoGateInserts.cfg を構成します。このファイルには、eventPrecedence insert を構成できる表が含まれています。

イベント・ゲートウェイ
: イベント・ゲートウェイ構成ファイル EventGatewaySchema.cfg で定義されている insert を使用して、config.precedence テーブルにデータを取り込みます。

イベント・ゲートウェイで構成されている eventPrecedence insert は、Tivoli Netcool/OMNIbus プローブ・ルール・ファイル内または Netcool/OMNIbus ナレッジ・ライブラリー 内で構成されている eventPrecedence insert をオーバーライドします。これにより、ネットワーク内で構成されている eventPrecedence insert をローカルでオーバーライドすることができます。

デフォルトのイベント・マップ:

Network Manager には、デフォルトのイベント・マップ・セットが用意されています。ここでは、使用可能なデフォルトのイベント・マップおよび各イベント・マップの仕組みについて説明し、さらに 3.9 イベント・マップがレガシー・イベント・マップを代行する方法を説明します。

デフォルトのイベント・マップ

次の表にデフォルトのイベント・マップを示します。

表 *18.* デフォルトのイベント・マップ

イベント・マップ	イベント・マップによって呼び出されるスティッチャー	イベント・マップの説明
EntityFailure	LookupIp	エンティティーを指定するために LocalNodeAlias で十分な場合、または追加データが入手できない場合にイベントを処理します。 予想される入力フィールドは `LocalNodeAlias` です。このフィールドはノード IP アドレスまたは DNS 名を含みます。
EntityManagedStateChange	関連のスティッチャーはなし	プラグインが、エンティティーの保守状態の変更 (管理状況が 0 または 1 に変更される場合) に対応できるようにします。このイベント・マップは、IP アドレスを持つシャーシまたはインターフェースのみに適用されます。イベントは、デフォルトでは使用されません。 予想される入力フィールドは `NmosEntityId` です。このフィールドはエンティティーの NCIM entityId を含みます。
EntityMibTrap	LookupEntPhysEntry	ENTITY MIB からのトラップを処理します。 予想される入力フィールドは以下のとおりです。 • `LocalNodeAlias`。このフィールドはノード IP アドレス、DNS 名、sysName または entityName を含みます。 • `LocalPriObj`。このフィールドは entPhysicalTable からの索引 (`'entPhysicalEntry.2'` など) を含みます。

表 18. デフォルトのイベント・マップ (続き)

イベント・マップ	イベント・マップによって呼び出されるスティッチャー	イベント・マップの説明
genericip-event	LookupMainNode	他のどのイベント・マップとも一致しないイベントを処理します。 **注:** このイベント・マップはキャッチオールとして提供されています。プラグインは、このイベント・マップへの関心を登録してはなりません。代わりに、対象のイベントを EntityFailure イベント・マップに渡す必要があります。 予想される入力フィールドは `LocalNodeAlias` です。このフィールドはノード IP アドレスまたは DNS 名を含みます。
ItnmHealthChk	関連のスティッチャーはなし	フェイルオーバー・プラグインによって、Network Manager 正常性検査イベントの処理に使用されます。 予想される入力フィールドは `Node` です。このフィールドは正常性検査の対象ドメインの名前を含みます。
ItnmLinkDownIfIndex	LookupIfEntry	インターフェースの `NmosEntityId` が設定されていない場合に、インターフェース・イベントが ifIndex によって識別されることを予想します。 予想される入力フィールドは以下のとおりです。 • `LocalNodeAlias`。このフィールドはノード IP アドレス、DNS 名、sysName または entityName を含みます。 • `LocalPriObj`。このフィールドは、`ifTable` テーブルからの `ifIndex` 値を `ifEntry.`*`ifIndex`* の形式で含みます。ここで、*ifIndex* は `ifIndex` 値 (`ifEntry.1` など) です。
ItnmMonitorEventNoRca	LookupEntityId	Network Manager ポーリング・エンジン ncp_poller によって生成された、ルート原因分析を実行してはならないイベントを処理するために使用されます。 予想される入力フィールドは `NmosEntityId` です。このフィールドはエンティティーの NCIM entityId を含みます。
ItnmStatus	関連のスティッチャーはなし	他のどのイベント・マップにも明示的に処理されない Network Manager 状況情報イベント (ItnmHealthCheck など) 用のキャッチオールのイベント・マップ。これらのイベントにアクションはとられません。 Network Manager 状況情報イベントについての詳細は、*IBM Tivoli Network Manager IP Edition* インストールと構成ガイド を参照してください。

表 18. デフォルトのイベント・マップ (続き)

イベント・マップ	イベント・マップによって呼び出されるスティッチャー	イベント・マップの説明
LinkDownIfDescr	LookupIfEntry	インターフェース・イベントが ifDescr 値によって識別されることを予想します。 予想される入力フィールドは以下のとおりです。 • LocalNodeAlias。このフィールドはノード IP アドレス、DNS 名、sysName または entityName を含みます。 • LocalPriObj。このフィールドは、ifTable テーブルからの ifDescr 値を ifEntry.*ifDescr* の形式で含みます。ここで、*ifDescr* は ifDescr 値 (ifEntry.FastEthernet0/1 など) です。
LinkDownIfIndex	LookupIfEntry	インターフェース・イベントが ifIndex 値によって識別されることを予想します。 予想される入力フィールドは以下のとおりです。 • LocalNodeAlias。このフィールドはノード IP アドレス、DNS 名、sysName または entityName を含みます。 • LocalPriObj。このフィールドは、ifTable テーブルからの ifIndex 値を ifEntry.*ifIndex* の形式で含みます。ここで、*ifIndex* は ifIndex 値 (ifEntry.1 など) です。
LinkDownIfName	LookupIfEntry	インターフェース・イベントが ifName 値によって識別されることを予想します。 予想される入力フィールドは以下のとおりです。 • LocalNodeAlias。このフィールドはノード IP アドレス、DNS 名、sysName または entityName を含みます。 • LocalPriObj。このフィールドは、ifTable テーブルからの ifName 値を ifEntry.*ifName* の形式で含みます。ここで、*ifName* は ifName 値 (ifEntry.Fa0/1 など) です。

表 18. デフォルトのイベント・マップ (続き)

イベント・マップ	イベント・マップによって呼び出されるスティッチャー	イベント・マップの説明
NbrFail	LookupNbrFailure	OSPF、LDP、および BGP 隣接変更イベントを処理します。 LookupNbrFailure スティッチャーは、必要なルックアップを実行するだけでなく、RCA プラグインによって使用される RemoteNodeEntityId 値の追加も行います。 予想される入力フィールドは以下のとおりです。 • LocalNodeAlias。このフィールドはノード IP アドレス、DNS 名、sysName または entityName を含みます。 • RemoteNodeAlias。このフィールドは隣接ノードの IP アドレスまたは DNS 名を含みます。 • LocalPriObj。このフィールドは、ifDescr 値を ifEntry.*ifDescr* の形式で含みます。ここで、*ifDescr* は ifDescr 値 (ifEntry.ifFastEthernet0/1 など) です。
OspfIfState	LookupOspfIfEntry	OSPF インターフェース・イベントを処理します。 予想される入力フィールドは以下のとおりです。 • LocalNodeAlias。このフィールドはノード IP アドレス (アドレスのないインターフェースのみが使用) を含みます。 • LocalPriObj。このフィールドは ospfIfTable からの索引を含みます。例えば以下のものです。 – ospfIfEntry.0.0.0.0.66。アドレスのない (IP 無番号) インターフェースの場合。 – ospfIfEntry.84.82.109.12.0。シリアル・インターフェースまたはイーサネット・インターフェースの場合。
PollFailure	LookupIp	Tivoli Netcool/OMNIbus ping プローブ・イベントなどの、特定の IP アドレスに対して生成されたイベントを対象とします。 予想される入力フィールドは NmosEntityId です。このフィールドはエンティティーの NCIM entityId を含みます。
PrecisionMonitorEvent	LookupEntityId	ITNM ポーリング・プログラムによって生成された、ルート原因分析を実行する必要のあるイベントの処理に使用されます。 予想される入力フィールドは NmosEntityId です。このフィールドはエンティティーの NCIM entityId を含みます。

表 *18.* デフォルトのイベント・マップ *(*続き*)*

イベント・マップ	イベント・マップによって呼び出されるスティッチャー	イベント・マップの説明
Reconfiguration	LookupMainNode	デバイスの部分的ディスカバリーを、リブート・イベント (イベント ID が NmosSnmpReboot のイベント) に基づき Disco プラグインを使用して自動的に起動できるようにします。 予想される入力フィールドは `LocalNodeAlias` です。このフィールドはノード IP アドレスまたは DNS 名を含みます。

レガシー・イベント・マップ

次の表に、レガシー・イベント・マップをリストし、各レガシー・イベント・マップを代行する 3.9 イベント・マップを示します。

表 *19.* レガシー・イベント・マップ

レガシー・イベント・マップ	以下の **3.9** イベント・マップによって処理
EntityIfDescr	LinkDownIfDescr
NbrFailIfDescr	NbrFail
NcpHealthChk	ItnmHealthChk
OSPFIfStateChange	OspfIfState
OSPFIfStateChangeIP	OspfIfState

3.9 イベント・マップがレガシー・イベント・マップを代行する方法

この例では、NbrFail 3.9 イベント・マップが NbrFailIfDescr レガシー・イベント・マップをどのように代行するかを説明します。

1. イベント ID が config.precedence テーブルにリストされていて、NbrFailIfDescr イベント・マップにマッピングされているイベントを受け取るとします。
2. HandledBy フィールドを使用して、NbrFail イベント・マップが NbrFailIfDescr イベント・マップを代行します。
3. イベントは、最初から NbrFail イベント・マップにマップされた場合とまったく同じ方法で処理されます。すなわち、以下のように処理されます。
   - LookupNbrFailure スティッチャーが呼び出されます。このスティッチャーは、config.eventMaps テーブル内の NbrFail イベント・マップ項目に関連する `Stitcher` フィールド内で参照されています。
   - イベントがプラグインに渡される前に、(イベントが当初マップされた NbrFailIfDescr ではなく) NbrFail イベント・マップのフィールドがイベントに付加されます。
   - (イベントが当初マップされた NbrFailIfDescr ではなく) NbrFail イベント・マップへの関心を登録していたプラグインが、イベントを受け取ります。

この例では、スティッチャーの言語に柔軟性があるため、両方のタイプのイベントを同じ方法で処理できます。レガシーの NbrFailIfDescr イベント・マップによって予想されている ifDescr が使用可能な場合、これが抽出されて LookupNbrFailure スティッチャー内で使用されます。

関連概念:

241 ページの『付録 F. Network Manager のイベント・カテゴリー』
Network Manager によって生成されるイベントは、2 つのカテゴリーに分類されます。1 つはモニターされているネットワークに関するイベント、もう 1 つは Network Manager のプロセスに関するイベントです。

イベント・ゲートウェイ・スティッチャー

イベント・ゲートウェイ・スティッチャーは、イベントをエンティティーと突き合わせ、トポロジー・ルックアップを実行し、取得したトポロジー・データを使用してイベント・データを強化します。

イベント・ゲートウェイ・スティッチャーは、`$NCHOME/precision/eventGateway/stitchers/`に格納されます。

スティッチャー言語については、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。

以下の 4 つのタイプのイベント・ゲートウェイ・スティッチャーがあります。

- 124 ページの『トポロジー・ルックアップ・スティッチャー』
- 129 ページの『データ抽出スティッチャー』
- 131 ページの『エンティティー取得スティッチャー』
- 133 ページの『イベント強化スティッチャー』

これに加え、デフォルトでは使用されないイベント・ゲートウェイ・スティッチャーもいくつか提供されています。 これらのスティッチャーは、スティッチャーを使用するイベント・ゲートウェイに追加できる追加機能の例として提供されています。 詳しくは、 136 ページの『デフォルトで使用されないスティッチャー』を参照してください。

イベント・マップによるスティッチャーの選択**:**

ここでは、イベント・マップによって特定のイベント・ゲートウェイ・スティッチャーを呼び出せるようにイベント・ゲートウェイを構成する方法を説明します。

特定のイベント・ゲートウェイ・スティッチャーを選択するためのイベント・マップの構成は、イベント・ゲートウェイの config.eventMaps テーブルを使用して行います。config.eventMaps テーブルは EventGatewaySchema.cfg 構成ファイル内で構成します。このファイルの場所は、`$NCHOME/etc/precision/EventGatewaySchema.cfg`です。

次の例では、EventGatewaySchema.cfg 構成ファイル内で config.eventMaps テーブルの一部を構成する方法を示しています。この例の insert は、次の表にリストされているイベント・マップを構成し、それぞれ示されているスティッチャーが呼び出されるようにします。

表 20. イベント・マップおよび選択されたスティッチャー

イベント・マップ	選択されたスティッチャー
PollFailure	IpLookup
ItnmMonitorEventNoRca	EntityIdLookup
PrecisionMonitorEvent	EntityIdLookup
LinkDownIfIndex	IfEntryLookup

特定のイベント・ゲートウェイ・スティッチャーを選択するためのイベント・マップの構成に関連するコード部分を、この例の以下の行にリストします。次の表では、この例の関連する行について説明します。このコードは config.eventMaps テーブルを参照します。

表 21. 着信イベント・フィルターに関連するコードの行

行番号	説明
1-10	IpLookup スティッチャーを選択するように、イベント・マップ PollFailure を構成します。
12-21	EntityIdLookup スティッチャーを選択するように、イベント・マップ ItnmMonitorEventNoRca を構成します。
23-32	EntityIdLookup スティッチャーを選択するように、イベント・マップ PrecisionMonitorEvent を構成します。
34-45	IfEntryLookup スティッチャーを選択するように、イベント・マップ LinkDownIfIndex を構成します。 これはトラップ・イベントであるため、イベントがフラップする可能性があります。フラップとは、デバイスまたはインターフェースがネットワークに接続されて切断されるという動作が、短時間の間に繰り返し行われる状態を言います。これが起こると、同じデバイスまたはインターフェースについて、問題イベントとクリア・イベントを交互に繰り返し受け取る事態が生じます。 EventCanFlap フラグを 1 に設定すると、RCA プラグインにこの状態が通知されます。 RCA プラグインは、このフラグが 1 に設定されたイベントが実際にフラップしているか、つまり、デバイスまたはインターフェースが連続的に接続 (アップ) 状態になったり切断 (ダウン) 状態になったりしているかを確認し、フラップしていれば、イベントが安定するまで待機します。 フラップできるイベントは、`EventCanFlap = 1` が設定されてイベント・ゲートウェイから渡されます。これらのイベントは、`TimedEscalation = 1` が指定されて mojo.events データベースに入れられ、そのまま 30 秒間置かれます。30 秒を過ぎると、TimedEventSuppression RCA スティッチャーは、30 秒以上経過し、なおかつ `TimedEscalation = 1` が設定されたイベントをすべて処理します。

```
1]    insert into config.eventMaps
2]    (
3]        EventMapName,
4]        Stitcher
5]    )
6]    values
7]    (
8]        "PollFailure",
9]        "IpLookup"
10]   );
11]
12]   insert into config.eventMaps
13]   (
14]       EventMapName,
15]       Stitcher
16]   )
17]   values
18]   (
19]       "ItnmMonitorEventNoRca",
20]       "EntityIdLookup"
21]   );
22]
23]   insert into config.eventMaps
24]   (
25]       EventMapName,
26]       Stitcher
27]   )
28]   values
29]   (
30]       "PrecisionMonitorEvent",
31]       "EntityIdLookup"
32]   );
33]
34]   insert into config.eventMaps
35]   (
36]       EventMapName,
37]       Stitcher,
38]       EventCanFlap
39]   )
40]   values
41]   (
42]       "LinkDownIfIndex",
43]       "IfEntryLookup",
44]       1
45]   );
```

関連資料:

272 ページの『config.eventMaps テーブル』
config.eventMaps テーブルには、イベントの処理方法を示すイベント・マップが含まれます。テーブルは、イベント・ゲートウェイによって処理される Tivoli Netcool/OMNIbus イベントのそれぞれのタイプに固有の情報を保持します。

イベント・ゲートウェイ・スティッチャーの説明:

イベント・ゲートウェイ・スティッチャーは 4 つのカテゴリーに分類されます。各カテゴリーのスティッチャーは、トポロジー・ルックアップおよびイベント・エンリッチのそれぞれの側面を担当します。

トポロジー・ルックアップ・スティッチャー:

これらはイベント・マップにリストされているスティッチャーです。 トポロジー・ルックアップ・スティッチャーは、未加工イベントを受け取って、トポロジー・ルックアップを実行し、何らかのイベント・エンリッチを実施します。 多くの場合、これらのスティッチャーは他のスティッチャーを利用してこれらのタスクを実行します。 例えば、データ抽出スティッチャーを使用してイベントから情報を抽出したり、エンティティー取得スティッチャーを使用してイベントを NCIM キャッシュ内のエンティティーと突き合わせたり、イベント・エンリッチ・スティッチャーを使用してイベントを強化したりします。

これらのスティッチャーは、トポロジー・マネージャー ncp_model によってブロードキャストされる、NCIM キャッシュ内のトポロジーをルックアップします。NCIM キャッシュについて、および NCIM キャッシュのデータ形式については、*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンスを参照してください。

トポロジー・ルックアップ・スティッチャーは、1 または 0 のブール値を返します。これらの戻り値には以下の意味があります。

戻り値 **1**
: 強化されたイベント・データをサブスクライブしているプラグインに渡します。 通常これは、トポロジー・ルックアップが正常に行われ、NCIM キャッシュ内でエンティティーが見つかったことを意味します。

戻り値 **0**
: イベント・データをどのプラグインにも渡しません。 通常これは、トポロジー・ルックアップが正常に行われず、NCIM キャッシュ内でエンティティーが見つからなかったことを意味します。

次の表にトポロジー・ルックアップ・スティッチャーを示します。

表 22. トポロジー・ルックアップ・スティッチャー

スティッチャー	説明	予想される引数	戻り
LookupEntityId.stch	イベントの NmosEntityId フィールドの値のみに基づいてエンティティーをルックアップします。 このスティッチャーは、Network Manager ポーリング・エンジン ncp_poller によって生成されるイベントのみが使用するためのものです。 ルックアップの結果に基づいてデフォルトのイベント強化を実行します。	なし。 イベント・マップから呼び出されます。	以下の値のいずれかを戻します。 • 1: エンティティーが NCIM キャッシュで検出されました。 強化されたイベント・データをサブスクライブしているプラグインに渡します。 • 0: エンティティーが NCIM キャッシュで検出されません。 強化されたイベント・データをサブスクライブしているプラグインに渡しません。
LookupEntPhysEntry	LocalPriObj フィールドの Entity MIB からの entPhysicalIndex データに基づいてエンティティーをルックアップします。 ルックアップの結果に基づいてデフォルトのイベント強化を実行します。	なし。 イベント・マップから呼び出されます。	以下の値のいずれかを戻します。 • 1: エンティティーが NCIM キャッシュで検出されました。 強化されたイベント・データをサブスクライブしているプラグインに渡します。 • 0: エンティティーが NCIM キャッシュで検出されません。 強化されたイベント・データをサブスクライブしているプラグインに渡しません。
LookupIfEntry.stch	イベント内のフィールド値に基づいて、デバイス上のインターフェースの索引項目をルックアップします。 このスティッチャーは、インターフェース上で発生するイベントによって使用されます。 ルックアップの結果に基づいてデフォルトのイベント強化を実行します。	なし。 イベント・マップから呼び出されます。	以下の値のいずれかを戻します。 • 1: エンティティーが NCIM キャッシュで検出されました。 強化されたイベント・データをサブスクライブしているプラグインに渡します。 • 0: エンティティーが NCIM キャッシュで検出されません。 強化されたイベント・データをサブスクライブしているプラグインに渡しません。

表 22. トポロジー・ルックアップ・スティッチャー (続き)

スティッチャー	説明	予想される引数	戻り
LookupIp.stch	IP アドレスまたは DNS 名を使用してエンティティーをルックアップします。 スティッチャーが検出するエンティティーはインターフェースまたはメイン・ノードのいずれかであることに注意してください。 ルックアップの結果に基づいてデフォルトのイベント強化を実行します。	なし。 イベント・マップから呼び出されます。	以下の値のいずれかを戻します。 • 1: エンティティーが NCIM キャッシュで検出されました。 強化されたイベント・データをサブスクライブしているプラグインに渡します。 • 0: エンティティーが NCIM キャッシュで検出されません。 強化されたイベント・データをサブスクライブしているプラグインに渡しません。
LookupMainNode.stch	メイン・ノード・デバイスを検索します。 イベントの NmosEntityId フィールドに値が存在する場合は、その値を使用してメイン・ノードのエンティティー ID が特定されます。 存在しない場合は、LocalNodeAlias フィールドの値にフォールバックします。 ルックアップの結果に基づいてデフォルトのイベント強化を実行します。	なし。 イベント・マップから呼び出されます。	以下の値のいずれかを戻します。 • 1: エンティティーが NCIM キャッシュで検出されました。 強化されたイベント・データをサブスクライブしているプラグインに渡します。 • 0: エンティティーが NCIM キャッシュで検出されません。 強化されたイベント・データをサブスクライブしているプラグインに渡しません。
LookupNbrFailure	IP アドレスまたは DNS 名をオプションのインターフェースの説明と一緒に使用して、エンティティーをルックアップします。 このスティッチャーは、イベントが関連するリモート・ノードもルックアップします。ルックアップの結果に基づいてデフォルトのイベント強化を実行します。	なし。 イベント・マップから呼び出されます。	以下の値のいずれかを戻します。 • 1: エンティティーが NCIM キャッシュで検出されました。 強化されたイベント・データをサブスクライブしているプラグインに渡します。 • 0: エンティティーが NCIM キャッシュで検出されません。 強化されたイベント・データをサブスクライブしているプラグインに渡しません。

表 22. トポロジー・ルックアップ・スティッチャー (続き)

スティッチャー	説明	予想される引数	戻り
LookupOspfIfEntr	ospfIfEntry データに基づいてインターフェースをルックアップします。ルックアップの結果に基づいてデフォルトのイベント強化を実行します。 このスティッチャーは、以下のいずれかの形式に基づいて OSPF データを確認します。 **ospfIfEntry.0.0.0.0.***ifIndex* この形式の例は `ospfIfEntry.0.0.0.0.66` です。この例では、抽出されたインターフェース索引の値は 66 です。 この形式は、IP 無番号インターフェースとも呼ばれている、アドレスのないインターフェースからのインターフェース・イベントに適用されます。 この形式は P2P シリアル・ポート・インターフェースによって使用されます。 **ospfIfEntry.***ipV4Adress***.0** この形式の例は `ospfIfEntry.84.82.109.12.0` です。この例では、抽出されたインターフェース索引の値は 84.82.109.12 という IP アドレスです。 この形式は、IP アドレスが割り当てられているインターフェース上の OSPF インターフェース・イベントに適用されます。 この形式はシリアル・インターフェースおよびイーサネット・インターフェースによって使用されます。	なし。 イベント・マップから呼び出されます。	以下の値のいずれかを戻します。 • 1: エンティティーが NCIM キャッシュで検出されました。 強化されたイベント・データをサブスクライブしているプラグインに渡します。 • 0: エンティティーが NCIM キャッシュで検出されません。 強化されたイベント・データをサブスクライブしているプラグインに渡しません。

例*: **LookupIp.stch*** スティッチャー*:*

このトピックでは、トポロジー・ルックアップ・スティッチャーの仕組みについて説明します。

LookupIp.stch スティッチャーは、IP アドレスまたは DNS 名を使用してエンティティーをルックアップします。スティッチャーが検出するエンティティーは、インターフェースまたはメイン・ノードのいずれかです。次の表では、このスティッチャーの主な要素について説明します。

表 *23. LookupIp.stch* スティッチャーの行ごとの説明

行番号	説明
3-7	このスティッチャーにはトリガーはありません。このスティッチャーは、PollFailure および EntityFailure イベント・マップによって自動的に呼び出されます。これらのイベント・マップは、このスティッチャーを呼び出す際に、関連付けられているイベントをスコープ内レコードとして指定します。
11	イベント (スコープ内レコード) に関連付けられているトポロジー・データを格納するために、entity という名前のレコードを作成します。
13	イベント内の NmosEntityId フィールドにアクセスして、このフィールドの値を nmosEntityId 変数にロードします。
14	GwEntityData() スティッチャー・ルールを使用して、nmosEntityId 変数の値に基づき、NCIM キャッシュ内のエンティティーの詳細をルックアップします。 GwEntityData() スティッチャー・ルール、およびその他のイベント・ゲートウェイ・スティッチャー・ルールについて詳しくは、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。
16-19	NmosEntityId フィールドが NULL の場合、このイベントが初めて発生したことを意味します。そのため、このイベントはこれまでイベント・ゲートウェイを通過したことがなく、強化もされていません。 NmosEntityId に代わる方法として、イベント・レコード内の LocalNodeAlias フィールドを使用して、影響を受けるエンティティーの ID を特定します。その後に、GwIpLookup() スティッチャー・ルールを使用して、LocalNodeAlias 変数の値に基づき、NCIM キャッシュ内のエンティティーの詳細をルックアップします。GwIpLookup() スティッチャー・ルール、およびその他のイベント・ゲートウェイ・スティッチャー・ルールについて詳しくは、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。
21	foundEntity 変数を使用してスティッチャーの戻り値を設定します。最初はこの変数の値を 0 に設定します。0 はエンティティーが見つかっていないことを想定した値です。
22-25	エンティティーが見つかった場合は、StandardEventEnrichment.stch スティッチャーを呼び出し、ルックアップによって取得したエンティティー・データを使用してイベントに対してイベント・エンリッチを実行します。スティッチャーの戻り値を 1 に設定します。
27	戻り値をイベント・ゲートウェイに渡します。

```
1]    UserDefinedStitcher
2]    {
3]        StitcherTrigger
4]        {
5]            // トリガーはありません。イベント・マップが、スコープ内レコードとして
6]    // イベントを指定して自動的にこれを呼び出すためです。
7]        }
8]
9]        StitcherRules
10]       {
11]           Record entity;
12]
13]           int nmosEntityId = eval(int, '&NmosEntityId');
14]           entity = GwEntityData( nmosEntityId );
15]
16]           if ( entity == NULL )
17]           {
18]               entity = GwIpLookupUsing( "LocalNodeAlias" );
19]           }
20]
21]           int foundEntity = 0;
22]           if ( entity <> NULL )
23]           {   ExecuteStitcher( "StandardEventEnrichment", entity );
24]               foundEntity = 1;
25]           }
26]
27]           SetReturnValue( foundEntity );
28]       }
29]   }
```

データ抽出スティッチャー*:*

これらのスティッチャーの唯一の目的は、標準形式の単一のイベント・データ・ストリングを取り出して解析し、単一値を抽出することです。この単一値がスティッチャーによって返されます。

次の表にデータ抽出スティッチャーを示します。

表 *24.* データ抽出スティッチャー

スティッチャー	説明	戻り
ExtractEntPhysIndex.stch	「entPhysicalEntry.*string identifier*」という形式の入力データ・ストリングから、インターフェース ID を表すテキスト値を抽出しようとします。 通常これは、LocalPriObj や LocalRootObj などのイベント・フィールドから値を抽出するために使用します。	インターフェース ID を表すストリング。通常は ifName、ifDescr、または ifAlias であると想定されます。

表 24. データ抽出スティッチャー (続き)

スティッチャー	説明	戻り
ExtractIfIndex.stch	「ifEntry.*numerical identifier*」という形式の入力データ・ストリングから、インターフェース ID を表す整数値を抽出しようとします。通常これは、LocalPriObj や LocalRootObj などのイベント・フィールドから ifIndex 値を抽出するために使用します。	整数値の索引
ExtractIfString.stch	「ifEntry.*string identifier*」という形式の入力データ・ストリングから、インターフェース ID を表すテキスト値を抽出しようとします。 通常これは、LocalPriObj や LocalRootObj などのイベント・フィールドから ifIndex 値を抽出するために使用します。	整数値の索引

例: ***ExtractIfString.stch*** スティッチャー:

このトピックでは、データ抽出スティッチャーの仕組みについて説明します。

ExtractIfString.stch スティッチャーは、`ifEntry.`*`string_identifier`* という形式のテキストのインターフェース ID を入力引数から抽出しようとします。ここで、*string_identifier* はテキストのインターフェース ID です。通常このメソッドは、LocalPriObj や LocalRootObj などのイベント・フィールドから ifIndex 値を抽出するために使用します。

表 25. *ExtractIfString.stch* スティッチャーの行ごとの説明

行番号	説明
3-11	このスティッチャーは、別のスティッチャー (通常はトポロジー・ルックアップ・スティッチャー) によって呼び出されます。
15	ifString 変数を NULL に初期化します。ifString 変数は、テキストのインターフェース ID の抽出操作の結果を保持します。
17	呼び出し側スティッチャーから入力引数を読み取り、これを ifInputStr 変数にロードします。
19	パターン・マッチングおよびデータ抽出操作の一部として使用する正規表現を指定します。
21-27	パターン・マッチングおよびデータ抽出操作を実行します。
29	抽出したストリングを呼び出し側スティッチャーに戻します。

```
1]    UserDefinedStitcher
2]    {
3]        StitcherTrigger
4]        {
5]            //
6]            // 以下の構文を使用して別のスティッチャーから呼び出されます。
7]            //
8]            // text ifString = "";
9]            // ifString = ExecuteStitcher( 'ExtractIfString', myStringField );
10]           //
11]       }
12]
13]       StitcherRules
14]       {
15]           text ifString = "";
16]
17]           text ifInputStr = eval(text, '$ARG_1');
18]
19]           text stringMatch = "^ifEntry¥.(¥S+)";
20]
21]           int stringMatchCount = MatchPattern( ifInputStr, stringMatch );
22]
23]           // フィールド全体で 1 回一致した場合にのみ、一致を認識します。
24]           if (stringMatchCount == 1 AND REGEX0 == ifInputStr )
25]           {
26]               ifString = eval(text, '$REGEX1');
27]           }
28]
29]           SetReturnValue( ifString );
30]       }
31]   }
```

エンティティー取得スティッチャー*:*

これらのスティッチャーは、事前定義のデータ (通常はイベントから抽出されたデータ) を受け取り、NCIM キャッシュ内の entityData テーブルから一致するエンティティーを取得しようとします。 一致するエンティティーが見つかった場合は取得した entityData レコードを返し、見つからなかった場合は NULL を返します。

次の表に、エンティティー取得スティッチャーを示します。

表 *26.* エンティティー取得スティッチャー

スティッチャー	説明	戻り
EntityFromEntPhysIndex.stch	入力としてテキスト形式のメイン・ノード名および整数形式の entPhysicalIndex 値を受け取った後、エンティティー MIB 内に示されている、entPhysicalIndex 値を持つ可能性のあるすべてのタイプのエンティティーを取得しようとします。	エンティティー MIB 内に示されている、entPhysicalIndex を持つ可能性のあるすべてのエンティティー・タイプ

表 26. エンティティー取得スティッチャー (続き)

スティッチャー	説明	戻り
EntityFromIfIndex.stch	入力としてテキスト形式のメイン・ノード名および整数形式の ifIndex 値を受け取った後、指定されたメイン・ノード上の、指定された ifIndex 値を持つインターフェースを取得しようとします。	指定されたメイン・ノード上の、指定された ifIndex を持つインターフェース
EntityFromIfString.stch	入力としてテキスト形式のメイン・ノード名、および整数形式の ifDescr 値、ifName 値、または ifAlias 値を受け取った後、指定されたストリングに一致する ifDescr 値、ifName 値、または ifAlias 値を持つ、指定されたメイン・ノード上のインターフェースを取得しようとします。	指定されたストリングに一致する ifDescr 値、ifName 値、または ifAlias 値を持つ、指定されたメイン・ノード上のインターフェース

例: ***EntityFromIfString.stch:***

このトピックでは、エンティティー取得スティッチャーの仕組みについて説明します。

EntityFromIfString.stch スティッチャーは、メイン・ノードの entityName およびストリング (インターフェースの ifName、ifDescr、または ifAlias と想定されます) に基づいてインターフェースをルックアップします。これは、NCIM キャッシュ内のインターフェース・エンティティー (見つかった場合) を返します。

表 27. *EntityFromIfString.stch* スティッチャーの行ごとの説明

行番号	説明
3-7	このスティッチャーは、別のスティッチャー (通常はトポロジー・ルックアップ・スティッチャー) によって呼び出されます。
11-12	呼び出し側スティッチャーから入力引数を読み取ります。
16-27	メイン・ノードの entityName およびストリング (インターフェースの ifName、ifDescr、または ifAlias と想定されます) に基づいて、インターフェースのエンティティー・データ・レコードを取得するための SQL 照会をセットアップします。
30	RetrieveSingleOQL() スティッチャー・ルールを使用して、照会を実行し、インターフェースのエンティティー・データ・レコードを取得します。RetrieveSingleOQL() スティッチャー・ルール、およびその他のイベント・ゲートウェイ・スティッチャー・ルールについて詳しくは、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。

表 27. *EntityFromIfString.stch* スティッチャーの行ごとの説明 (続き)

行番号	説明
32	エンティティーのルックアップ結果を呼び出し側スティッチャーに戻します。

```
1]    UserDefinedStitcher
2]    {
3]        StitcherTrigger
4]        {
5]            // これは他のスティッチャーから明示的に呼び出されるため、
6]    // トリガーはありません。
7]        }
8]
9]        StitcherRules
10]       {
11]           text mainNodeEntityName = ARG_1;
12]           text ifString = ARG_2;
13]
14]           Record entity;
15]
16]           text ifStringQuery =
17]                   "select * from ncimCache.entityData
18]                    where
19]                    ENTITYTYPE = 2
20]                    および
21]                    BASENAME = eval(text, '$mainNodeEntityName')
22]                    および
23]                    ( interface->IFNAME = eval(text, '$ifString')
24]                      or
25]                      interface->IFDESCR = eval(text, '$ifString')
26]                      or
27]                      interface->IFALIAS = eval(text, '$ifString') );";
28]
29]
30]           entity = RetrieveSingleOQL( ifStringQuery );
31]
32]           SetReturnValue( entity );
33]       }
34]   }
```

イベント強化スティッチャー:

これらのスティッチャーは、他のスティッチャーが取得したトポロジー・データを使用してイベントを強化します。

次の表にイベント強化スティッチャーを示します。

表 28. イベント強化スティッチャー

スティッチャー	説明	予想される引数	戻り
StandardEventEnrichment.stch	一部のプラグインが使用する予定のイベント・フィールド、および ObjectServer のイベントを更新するためにフィードバックされるフィールドにデータを取り込んで、デフォルトのイベント・エンリッチを実行します。	最上位のスコープ内イベントと一致したエンティティー	戻り値はありません。
EntityNotFound.stch	デフォルトでは、イベント・ゲートウェイによって設定されるフィールドは以下のものです。 • NmosObjInst • NmosSerial • NmosCauseType このスティッチャーは、以前にこのドメインにイベントが割り当てられたものの、一致するエンティティーが見つからない場合、これらの基本フィールドをリセットします。	なし	戻り値はありません。

例: ***StandardEventEnrichment.stch:***

このトピックでは、イベント・エンリッチ・スティッチャーの仕組みについて説明します。

StandardEventEnrichment.stch は標準のイベント・エンリッチを実行します。このスティッチャーは、プラグインが使用できると予想するイベント・フィールド (entityType など)、および ObjectServer でイベントを更新するためにイベント・ゲートウェイから直接フィードバックされるフィールドに取り込みます。ObjectServer の alerts.status テーブルを更新することを許可されているフィールドは、発信フィールド・フィルターにリストされているフィールドのみです。発信フィールド・フィルターは EventGatewaySchema.cfg 構成ファイル内の nco2ncp テーブルの FieldFilter セクションで定義されます。例えば、entityType および entityName フィールドはプラグインが使用するためにイベントに追加されますが、これらのフィールドは発信フィールド・フィルターを通過しないため、実際には ObjectServer でイベントを強化しません。

表 29. *StandardEventEnrichment.stch* スティッチャーの行ごとの説明

行番号	説明
3-8	このスティッチャーは、別のスティッチャー (通常はトポロジー・ルックアップ・スティッチャー) によって呼び出されます。
12	トポロジー・ルックアップ操作からエンティティー・データ・レコードを読み取ります。
13	イベントの強化に使用されるフィールドを保持するためのレコードを宣言します。
15-19	トポロジー・ルックアップ操作によって取得した値を使用して変数を初期化します。
19	GwManagedStatus() スティッチャー・ルールを使用して、エンティティーの管理状況を取得します。 GwManagedStatus() スティッチャー・ルール、およびその他のイベント・ゲートウェイ・スティッチャー・ルールについて詳しくは、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。
21-26	イベントの強化に使用されるレコード内のフィールド値を設定します。
28	GwEnrichEvent() スティッチャー・ルールを使用して、イベント内のフィールドを更新します。GwEnrichEvent() スティッチャー・ルール、およびその他のイベント・ゲートウェイ・スティッチャー・ルールについて詳しくは、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。

```
1]    UserDefinedStitcher
2]    {
3]        StitcherTrigger
4]        {
5]            // トリガーはありません。このスティッチャーは、スコープ内レコードと
6]               してイベントを指定し、
7]            // 単一引数としてエンティティーを指定して
8]    // 他のスティッチャーから呼び出されます。
9]        }
10]
11]       StitcherRules
12]       {
13]           Record entity = ARG_1;
14]           Record enrichedFields;
15]
16]           int entityType = @entity.entityData.ENTITYTYPE;
17]           text entityName = @entity.entityData.ENTITYNAME;
18]           int entityId = @entity.entityData.ENTITYID;
19]           int mainNodeId = @entity.entityData.MAINNODEENTITYID;
20]           int managedStatus = GwManagedStatus( entityId );
21]
22]           @enrichedFields.EntityType = entityType;
23]           @enrichedFields.EntityName = entityName;
24]           @enrichedFields.NmosDomainName = eval(text, '$DOMAIN_NAME');
25]           @enrichedFields.NmosEntityId = entityId;
26]           @enrichedFields.NmosManagedStatus = managedStatus;
27]           @enrichedFields.NmosObjInst = mainNodeId;
28]
29]           GwEnrichEvent( enrichedFields );
30]       }
      }
```

関連タスク:

182 ページの『例: メイン・ノード・デバイスのロケーションによるイベントの強化』
イベントに関連付けられているメイン・ノード・デバイスのロケーションがイベントのフィールドに追加されるように、イベント・エンリッチを構成することができます。

184 ページの『例: インターフェース名によるイベントの強化』
すべてのインターフェース・イベントに関してイベントが発生したインターフェースの名前がそのイベントのフィールドに追加されるように、イベント・エンリッチを構成することができます。

デフォルトで使用されないスティッチャー:

これらのスティッチャーは、スティッチャーを使用するイベント・ゲートウェイに追加できる追加機能の例として提供されています。 これらのスティッチャーは他のスティッチャーから実行できます。

次の表に、デフォルトで使用されないイベント・ゲートウェイ・スティッチャーを示します。

表 30. デフォルトで使用されないスティッチャー

スティッチャー	説明	予想される引数	戻り
EntityFromAtmIfDescr.stch	このスティッチャーは、NCIM キャッシュ内で関連付けられているトポロジーをルックアップする前に、(通常は ExtractIfString スティッチャーを使用して) イベントから抽出されたデータをどのように操作できるかを示します。この例では、標準サフィックスが欠落した、短縮されたインターフェースの説明を持つイベントが生成されます。 このサフィックスは、トポロジーをルックアップする前に追加されます。	メイン・ノード名 (テキスト) `-atm subif` サフィックスが欠落した `ifDecr` (テキスト)	イベントのインターフェースの説明と一致する `ifDescr` を持つ、指定されたメイン・ノード上のインターフェースに、事前定義の `-atm subif` ストリングを連結したもの。

表 30. デフォルトで使用されないスティッチャー (続き)

スティッチャー	説明	予想される引数	戻り
RetrieveAlertDetails.stch	Object Server の alert.details テーブルを調べて現在のスコープ内イベントに関連するデータがあるかどうかを確認し、データが見つかった場合はそれをスコープ内イベントに追加します。この例では、この操作が無選別に行われ、データのフィルター処理は実行されないことに注意してください。 さらに、この例では、追加データがあるかどうかについて Object Server を照会する方法を示すテンプレート例も提供されます。	なし	戻り値はありません

例: デフォルトでの Tivoli Netcool/OMNIbus トラップ・イベントの強化

ここでは、Tivoli Netcool/OMNIbus イベントがイベント・ゲートウェイを通過するときにどのように処理されるかを説明します。

イベント・ゲートウェイはオブジェクト・サーバーから Tivoli Netcool/OMNIbus リンクダウン・トラップ・イベントを受け取ります。このイベントは Tivoli Netcool/OMNIbus トラップ・プローブから発信されます。したがって、このイベントの発信元は Network Manager の外部になります。この例では、このイベントがイベント・ゲートウェイを通過するときにどのように処理されるかを説明します。

このプロセスの手順は以下のとおりです。

1. イベント・ゲートウェイはオブジェクト・サーバーからリンクダウン・トラップ・イベントを受け取ります。このイベントのイベント ID は `SNMPTRAP-LinkDown` です。
2. イベントに着信イベント・フィルターが適用されます。

   このフィルターはイベントの LocalNodeAlias フィールドを調べます。LocalNodeAlias フィールドは空でないため、イベントはフィルターを通過し、ステップ 3 に進みます。
3. イベント・ゲートウェイは、イベント内の Severity フィールド、Tally フィールド、および Type フィールドに基づいて、イベントに状態を割り当てます。このリンクダウン・トラップ・イベントには、以下に示す重大度と合計の情報が含まれています。
   - 重大度は非ゼロ

- 合計は 1
- タイプは「問題」

この情報に基づいて、イベント・ゲートウェイはこのイベントに「Occurred」という状態を割り当てます。これは問題イベントであり、RCA の候補です。

4. イベントにデフォルトの着信フィールド・フィルターが適用されます。このフィールド・フィルターは、イベント・ゲートウェイの処理に関与していない alerts.status フィールドをフィルターで除外し、以下のフィールドのみを通過させます。
   - Acknowledged
   - AlertGroup
   - EventId
   - FirstOccurrence
   - LastOccurrence
   - LocalNodeAlias
   - LocalPriObj
   - LocalRootObj
   - Manager
   - NmosCauseType
   - NmosDomainName
   - NmosEntityId
   - NmosEventMap
   - NmosManagedStatus
   - NmosObjInst
   - NmosSerial
   - Node
   - RemoteNodeAlias
   - EventId
   - Serial
   - ServerName
   - Severity
   - Summary
   - SuppressEscl
   - Tally
   - タイプ
5. イベント・ゲートウェイは、使用するイベント・マップを決定して、このイベントの処理方法を決定します。イベント・マップはイベントの処理方法を定義します。同時に、イベントに数値の優先順位値が関連付けられます。

この例のイベントのイベント ID は SNMPTRAP-LinkDown です。このイベント ID を持つイベントの処理方法を定義する insert は、Netcool/OMNIbus ナレッジ・ライブラリー 構成ファイルの NcoGateInserts.cfg で定義され、その形式は次のとおりです。

```
insert into config.precedence
(
    Precedence,
    EventMapName,
    NcoEventId
)
values
(
    910,
    "LinkDownIfIndex",
    "SNMPTRAP-LinkDown"
);
```

この insert は、SNMP リンクダウン・トラップ・イベントを次のように処理することをイベント・ゲートウェイに指示します。

- イベントにイベント・マップ LinkDownIfIndex を適用します。
  - このイベント・マップは、Tivoli Netcool/OMNIbus mttrapd プローブからのリンクアップ・イベントおよびリンクダウン・イベントを対象とします。これらのイベントはすべて、alerts.status の LocalPriObj フィールドに保持されている ifIndex 値を使用して、このトラップの発信元であるインターフェースを識別します。
  - RCA プラグインは、LinkDownIfIndex イベント・マップによって処理されるイベントをサブスクライブします。このため、このイベントは RCA プラグインに渡されます。
- RCA に対してイベントを送信するときは、イベントで 910 の優先順位値を使用します。

6. イベント・ゲートウェイは選択されたイベント・マップを使用して、トポロジー・ルックアップを実行するために呼び出すスティッチャーを決定します。

   選択されたイベント・マップは LinkDownIfIndex です。EventGatewaySchema.cfg 構成ファイルで、このイベント・マップに次の insert が関連付けられます。

```
insert into config.eventMaps
(
    EventMapName,
    Stitcher,
    EventCanFlap
)
values
(
    "LinkDownIfIndex",
    "LookupIfEntry",
    1
);
```

この insert は、以下のアクションを実行するようにイベント・ゲートウェイに指示します。

- LookupIfEntry スティッチャーを使用してトポロジー・ルックアップを実行します。

LookupIfEntry は、イベント内のフィールド値に基づいて、デバイス上のインターフェースの索引項目をルックアップします。イベント内のフィールドから抽出したインターフェース索引の値に基づいて、スティッチャーはエンティティーとインターフェース・データから構成される NCIM キャッシュから行を取得します。イベント・エンリッチを実行する別のスティッチャーが呼び出されます。

- EventCanFlap フラグを 1 に設定して、関連付けられているデバイスまたはインターフェースが引き続き接続 (アップ) 状態になったり切断 (ダウン) 状態になったりする可能性があることを RCA プラグインに通知します。

7. 強化されたイベント・データは発信フィールド・フィルターによってフィルター処理され、イベント・ゲートウェイ・キューに入れられます。
8. イベントが RCA プラグインに渡されます。
9. RCA によってさらなるイベント・エンリッチが行われた後、イベントはイベント・ゲートウェイ・キューに入れられます。

イベント・ゲートウェイ・プラグイン

イベント・ゲートウェイ・プラグインは、イベント・ゲートウェイ・プロセスのモジュールであり、強化されたイベントをイベント・ゲートウェイから受け取って、そのイベントに対してさらなるイベント・エンリッチまたはその他のアクションを実行します。

関連概念:

154 ページの『根本原因の分析』
ルート原因分析 (RCA) は、1 つ以上のデバイスのアラートの根本原因を判別するプロセスです。ルート原因分析 (RCA) プラグインはイベント・ゲートウェイから強化されたイベントのサブセットを受け取り、どのイベントが根本原因であり、どのイベントが副次であるかを判別します。 RCA は、ネットワークを介するトラフィックのルーティングに影響を与えるイベントのみを受信します。

プラグインの説明

ここでは、各イベント・ゲートウェイ・プラグインの仕組みについて説明します。

次の表にイベント・ゲートウェイ・プラグインを示します。これらの各プラグインは、イベント・ゲートウェイから強化されたイベントを受け取り、そのイベントをさらに強化するか、他の何らかのアクションを実行します。

注: 以下のプラグインがデフォルトで有効になっています。

- 適応ポーリング
- フェイルオーバー
- RCA
- すべての SAE (サービスに影響を与えるイベント) プラグイン

表 31. イベント・ゲートウェイ・プラグイン

プラグイン	説明
適応ポーリング	関連するイベントとエンティティー・データのサブセットを ncmonitor.activeEvent テーブルに書き込みます。 これにより、イベント・データ (アラート・ビュー) に基づいてネットワーク・ビューを作成できます。 ポーリング・ポリシーはネットワーク・ビューに基づいてスコープされるため、ネットワーク・アラートに基づいてポーリングを定義できます。 これらが適応ポーリングと呼ばれる理由は、ポーリングがネットワーク問題の状態に基づいて開始されるためです。
Disco	イベント・ゲートウェイからリブート・イベント (トラップ) を受け取り、それらのイベントに基づいて部分的ディスカバリーを開始します。 デフォルトでは、このプラグインはリブートが行われたことを示すイベントのみを受け取ります。 **注:** デフォルトでは、このプラグインによって開始される部分的ディスカバリーはリブート・イベントに制限されます。 部分的ディスカバリーはリソースを多量に使用するプロセスであるため、受け取ったすべてのイベント、特に数量を多く受け取るイベント (リンクダウンなど) に対して部分的ディスカバリーがトリガーされると、問題が発生する可能性が高くなります。
フェイルオーバー	イベント・ゲートウェイから Network Manager 正常性検査 (ItnmHealthChk) イベントを受け取り、それらのイベントを仮想ドメイン・プロセスに渡します。仮想ドメイン・プロセスは、フェイルオーバーを開始するかどうかをそのイベントに基づいて決定します。
ルート原因分析 (RCA)	RCA スティッチャー内でコーディングされているルールが、イベント内のデータおよびディスカバーされたトポロジーに基づいて、他のイベントによって引き起こされるイベント、または他のイベントを引き起こすイベントを識別しようとします。
サービスに影響を与えるイベント (SAE)	デフォルトでは、SAE プラグインは以下の 3 つのタイプの SAE を生成します。 • MPLS VPN: MPLS VPN のサービスに影響を与えるイベントを生成します。 • IP パス: IP パスのサービスに影響を与えるイベントを生成します。 • 汎用 Network Manager サービス: カスタム・サービスと関連付けられているデバイス上でイベントが発生したときに生成される合成イベントに対して構成することができます。
zNetView	IBM Tivoli NetView for z/OS® によって使用される追加のカスタム alerts.status フィールドに取り込みます。 **注:** 最初に以下のカスタム・フィールドを alerts.status テーブルに追加する必要があります。 • NmosClassName • NmosEntityType

関連概念:

154 ページの『根本原因の分析』
ルート原因分析 (RCA) は、1 つ以上のデバイスのアラートの根本原因を判別するプロセスです。ルート原因分析 (RCA) プラグインはイベント・ゲートウェイから強化されたイベントのサブセットを受け取り、どのイベントが根本原因であり、どのイベントが副次であるかを判別します。 RCA は、ネットワークを介するトラフィックのルーティングに影響を与えるイベントのみを受信します。

関連タスク:

191 ページの『プラグインの有効化と無効化』
プラグインを有効または無効にするには、ncp_gwplugins.pl スクリプトを使用します。プラグインごとにスクリプトを個別に実行します。

192 ページの『プラグイン情報のリスト』
イベント・ゲートウェイ・プラグインに関する情報をリストすることができます。例えば、各プラグインがサブスクライブしているイベント・マップとイベント状態をリストすることができます。

193 ページの『イベント・マップ・サブスクリプションの変更』
プラグインがサブスクライブしているイベント・マップを変更することができます。例えば、新しいイベント・マップを追加し、そのイベント・マップによって処理されるイベントに対してシステムによる RCA を実行する場合は、そのイベント・マップを RCA プラグインのサブスクリプション・リストに追加する必要があります。

195 ページの『プラグイン構成パラメーターの設定』
`ncp_gwplugins.pl` スクリプトを使用して、イベント・ゲートウェイ・プラグインのオプションの構成パラメーターを設定することができます。

61 ページの『第 8 章 適応ポーリングの管理』
適応ポーリングは、ネットワークのイベントに動的に反応します。さまざまなネットワーク問題のシナリオを管理する適応ポーリングを作成することが可能です。

適応ポーリング・プラグイン

ここでは、プラグインの前提条件、適応ポーリング・プラグインが activeEvent テーブルのフィールドにデータを取り込む方法、およびこのプラグインに関連する構成の詳細について説明します。 activeEvent テーブルは、NCMONITOR スキーマ内にあります。

適応ポーリング・プラグインは、ObjectServer でイベントがクリアまたは削除されたときに、activeEvent テーブルから行を削除します。

必須フィールド

このプラグインでは、イベント・ゲートウェイによって渡されるイベントに、以下のフィールドが取り込まれていることが想定されています。

- Acknowledged
- AlertGroup
- EventId
- FirstOccurrence
- LastOccurrence
- LocalPriObj
- NmosCauseType
- NmosSerial
- NmosEntityId
- Serial
- Severity
- SuppressEscl
- Tally

activeEvent テーブル内のイベント

activeEvent テーブルには、トポロジー内のエンティティーに一致していて、なおかつ以下の条件を満たす、アクティブな問題イベントのみが含まれます。

- イベントがアクティブである。これは、イベントがクリアされておらず、フィールド条件で `Severity > 0` という関係によって表されることを意味します。

- イベントが問題イベントである。alerts.status フィールドのタイプの値には、Problem、More Severe、または Less Severe があります。
- イベントがエンティティーと一致している。イベント・ゲートウェイはメイン・ノードを識別しています。これは、フィールド条件で、NmosObjInst > 0 という関係によって表されます。

activeEvent テーブルのフィールド

次の表に、適応ポーリング・プラグインによってデータが取り込まれる activeEvent テーブルのフィールドをリストします。これらのフィールドを利用するアラート・ビューの作成例については、*IBM Tivoli Network Manager IP Edition* ネットワーク可視化セットアップ・ガイドを参照してください。

表 32. 適応ポーリング・プラグインによってデータが取り込まれる *activeEvent* テーブルのフィールド

プラグイン	説明
Acknowledged	オペレーターによってイベントが確認されたかどうかを示します。
AlertGroup	イベントの発信元を識別します。
domainMgrId	影響を受けるデバイスの所属ドメインを識別する固有の整数。
entityId	イベントの NmosEntityId。他の NCIM トポロジー・データベース表との整合性を確保して、GUI 機能を使いやすくするために、このテーブルではこのフィールドの名前が entityId になっています。
EventId	イベント・タイプの名前。このフィールドに基づいて、特定の障害タイプに対する 2 次ポーリングを開始できます。
FirstOccurrence	このイベントが作成された、またはポーリングが開始した (1970 年 1 月 1 日午前 0 時からの) 時間 (単位は秒)。
LastOccurrence	このイベントが最終更新された時間。
LocalPriObj	イベントによって参照される 1 次オブジェクト。管理対象オブジェクト・インスタンスの識別に使用されます。
NmosCauseType	ルート原因分析の結果を格納し、ルート原因イベントを識別できるようにします。
NmosSerial	ルート原因分析中にイベントが抑止された場合、抑止イベントの Serial フィールドの値がこのフィールドに示されます。
Serial	単一 ObjectServer のコンテキストにおけるイベントの固有 ID。このフィールドは、このテーブル用のキーを生成できるようにするためにこのテーブルに格納されます。
ServerName	ObjectServer の固有の名前。複数の ObjectServers を持つシステムでは、イベントを一意的に識別するためにこれが必要です。これは、テーブル用のキーを生成できるようにすることのみを目的として、テーブルに取り込まれています。
Severity	イベントの重大度。 **注:** 重大度ゼロのイベントはリストされません。それらのイベントは、アラートが解決済みであることを表しているため、アラート・ビューには必要なく、アラート・ビューをスコープとして使用するポーリング・ポリシーにも必要ありません。
SuppressEscl	アラートの抑止、またはエスカレーションに使用されます。抑止レベルはオペレーターがアクティブ・イベント・リストから手動で選択します。
Tally	イベントが発生した回数。これにより、単発的な ping 障害などの散発的なイベントをフィルターで除外することができます。

プラグインの構成

このプラグインは、始動時、またはイベント・ゲートウェイの再同期時 (SIGHUP 時、あるいはフェイルオーバー時かフェイルバック時) に、オブジェクト・サーバーの alerts.colors テーブルおよび alerts.conversions テーブル内の値に基づいて、alertColors テーブルおよび alertConversions テーブルへのデータの取り込みも行います。gwPluginConf テーブルでは、オプションで以下のパラメーターを設定できます。このテーブルは NCMONITOR スキーマ内にあります。

表 *33.* 適応ポーリング・プラグインのオプション構成

パラメーター名	値	目的	デフォルト
CopyAlertTablesAtStartup	alertColors テーブルおよび alertConversions テーブルにデータを取り込むかどうかを示します。可能な値は次のとおりです。 • True • False	これにより、複数のドメインの実行時に問題が発生した場合、単一のドメインによってこれらのテーブルにデータを取り込むことができます。	True
ActiveEventUpdateInterval	activeEvent テーブルの更新間隔 (秒単位) です。	テーブルの更新はトランザクションで行われ、DB サーバーの負荷を最小限に抑えるように試みられます。この間隔は、これらのトランザクションがコミットされる期間を示します。	5

関連タスク:

61 ページの『第 8 章 適応ポーリングの管理』
適応ポーリングは、ネットワークのイベントに動的に反応します。さまざまなネットワーク問題のシナリオを管理する適応ポーリングを作成することが可能です。

関連資料:

280 ページの『ncmonitor 内の ncp_g_event プラグイン・データベース表』
ここでは、ncmonitor データベース内にあるイベント・ゲートウェイ構成表と、各表に含まれている情報の種類について説明します。 これらの表のほとんどは、イベント・ゲートウェイのプラグイン構成に関係します。

Disco プラグイン

ここでは、このプラグインの基本的な操作、プラグインの前提条件、およびこのプラグインに関連する構成の詳細について説明します。

プラグインの操作

Disco プラグインは再構成イベント・マップをサブスクライブします。デフォルトでは、イベント ID が NmosSnmpReboot のイベントのみが再構成イベント・マップによって処理されます。これらのイベントは Network Manager rebootDetection ポーリング・ポリシーに基づいており、デバイスでリブートが行われたことを示します。他のトラップを処理するように Disco プラグインを構成するには、関連イベントが再構成イベント・マップによって処理されるように構成します。

次の図では、Disco プラグインの操作を簡単に要約しています。

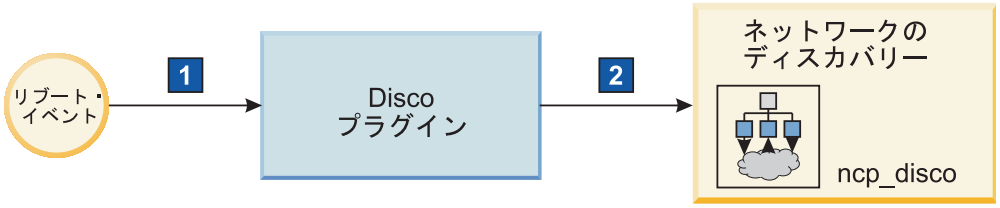


図 *4. Disco* プラグインの操作

1 リブート・イベントを受け取ります。
リブート・イベントがイベント・ゲートウェイから Disco プラグインに渡されます。

2 部分的なディスカバリー要求が生成されます。
Disco プラグインは、ディスカバリー・エンジン ncp_disco の finders.returns テーブルに対する適切な OQL ステートメントにリブート・イベントを変換します。

3 再ディスカバリー要求がディスカバリー・エンジン **ncp_disco** に送信されます。 OQL ステートメントが実行されて、関連する 1 つ以上のデバイスの部分的なディスカバリーが起動されます。

必須フィールド

このプラグインでは、イベント・ゲートウェイによって渡されるイベントに、以下のフィールドが取り込まれていることが想定されています。

- NmosObjInst

プラグインの構成

ncmonitor.gwPluginConf テーブルで次のパラメーターを設定する必要があります。

表 *34.* 適応ポーリング・プラグインのオプション構成

パラメーター名	値	目的	デフォルト
StitcherSubDir	このプラグイン用のスティッチャーが含まれている、`$NCHOME/precision/eventGateway/stitchers/` ディレクトリー内のサブディレクトリーの名前。	このディレクトリーの名前を指定することで、Disco プラグインのみがそのスティッチャーを解析できるようになります。	`Disco`

ncmonitor.gwPluginConf テーブルでは、オプションで以下のパラメーターを設定できます。

表 35. 適応ポーリング・プラグインのオプション構成

パラメーター名	値	目的	デフォルト
StartupStitcher	初期化時に実行する、`$NCHOME/precision/eventGateway/stitchers/` 内のサブディレクトリーにあるスティッチャーの名前。	開始スティッチャー名が指定されている場合、開始時、SIGHUP 時、またはフェイルオーバー時かフェイルバック時に、このスティッチャーが引数なしで実行されます。	なし
SchemaFile	初期化時に解析する、`$NCHOME/etc/precision/` 内のスキーマ・ファイルの名前。	スキーマ・ファイル名が指定されている場合、開始スティッチャーが実行される前の、開始時、SIGHUP 時、またはフェイルオーバー時かフェイルバック時に、このファイルが解析されます。	なし

関連資料:

280 ページの『ncmonitor 内の ncp_g_event プラグイン・データベース表』
ここでは、ncmonitor データベース内にあるイベント・ゲートウェイ構成表と、各表に含まれている情報の種類について説明します。 これらの表のほとんどは、イベント・ゲートウェイのプラグイン構成に関係します。

フェイルオーバー・プラグイン

この情報を使用して、プラグインの操作、およびプラグインに関連する構成の詳細について説明します。

プラグインの操作

このプラグインに関連する構成情報はありません。次の図は、フェイルオーバー・プラグインの操作を簡潔に要約したものです。

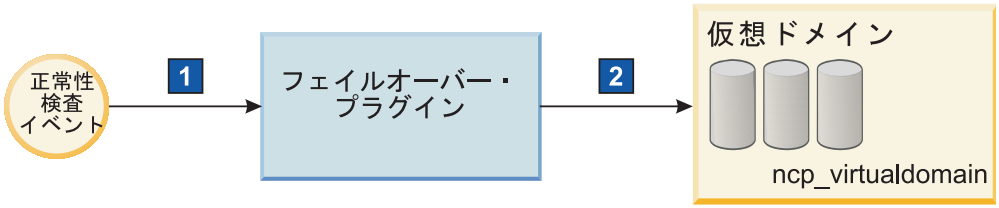


図 5. フェイルオーバー・プラグインの操作

1 **Network Manager** ヘルス・チェック・イベントを受け取る
Network Manager ヘルス・チェック・イベントがイベント・ゲートウェイからフェイルオーバー・プラグインに渡されます。

2 フェイルオーバー要求が生成される
フェイルオーバー・プラグインは、Network Manager ヘルス・チェック・イベントを仮想ドメイン ncp_virtualdomain の state.domains テーブル用の適切な OQL ステートメントに変換します。

3 フェイルオーバー要求が仮想ドメイン **ncp_virtualdomain** に送られる OQL ステートメントが実行され、フェイルオーバーまたはフェイルバックが開始されます。

必須フィールド

このプラグインは、イベント・ゲートウェイによって渡されるイベントのノード・フィールドに、影響対象の Network Manager ドメインの名前が取り込まれることを想定します。

PostNCIMProcessing プラグイン

Fix Pack 4

PostNCIMProcessing プラグインは、NCIM データベースの更新後に必要なスティッチャーを実行します。デフォルトでは、このプラグインは、トポロジー更新イベントを受け取ると、単一の集約ドメインへの複数のドメインのスティッチをトリガーします。

プラグインの操作

PostNCIMProcessing プラグインは ItnmStatus イベント・マップをサブスクライブします。このプラグインのプロセス・フローは、以下の手順で説明するとおりです。

1. イベント・ゲートウェイが ItnmStatus イベントを受け取ります。
2. イベント・ゲートウェイが PostNCIMProcessing プラグインを呼び出します。
3. PostNCIMProcessing プラグインが、イベントのタイプを検査する PostNcimProcessing スティッチャーを実行します。
4. イベントのタイプが ItnmTopologyUpdate の場合、およびクロスドメイン・スティッチが有効になっている場合は、PostNCIMProcessing スティッチャーが AggregationDomain スティッチャーを実行します。
5. ディスカバリーが進行中でないことを AggregationDomain スティッチャーが検査してから、他の集約ドメイン・スティッチャーを実行します。これにより、ディスカバーされたドメインが単一の集約ドメインにスティッチされます。

必須フィールド

このプラグインでは、イベント・ゲートウェイによって渡されるイベントに、以下のフィールドが取り込まれていることが想定されています。

- NmosObjInst

プラグインの構成

ncmonitor.gwPluginConf テーブルで次のパラメーターを設定する必要があります。

表 36. *PostNCIMProcessing* プラグインの必須構成

パラメーター名	値	目的	デフォルト
StitcherSubDir	このプラグインのスティッチャーを格納する $NCHOME/precision/eventGateway/stitchers/ 内のディレクトリーの名前。	NCIM 処理スティッチャーをデフォルト・ディレクトリー PostNcimProcessing 内に格納しない場合に、スティッチャーを格納するディレクトリーを指定します。 **重要:** このディレクトリーが存在すること、およびスティッチャーを格納していることを確認してください。このサブディレクトリーが存在しない場合、プラグインは開始されません。	PostNcimProcessing

ncmonitor.gwPluginConf テーブルでは、オプションで以下のパラメーターを設定できます。

表 37. *Disco* プラグインのオプション構成

パラメーター名	値	目的	デフォルト
StartupStitcher	初期化時に実行する、$NCHOME/precision/eventGateway/stitchers/ 内のサブディレクトリーにあるスティッチャーの名前。	開始スティッチャー名が指定されている場合、開始時、SIGHUP 時、またはフェイルオーバー時かフェイルバック時に、このスティッチャーが引数なしで実行されます。	なし
SchemaFile	初期化時に解析する、$NCHOME/etc/precision/ 内のスキーマ・ファイルの名前。	スキーマ・ファイル名が指定されている場合、開始スティッチャーが実行される前の、開始時、SIGHUP 時、またはフェイルオーバー時かフェイルバック時に、このファイルが解析されます。	なし

SAE プラグイン

SAE プラグインは、MPLS VPN および IP パスのサービスに影響を与えるイベントを生成します。

デフォルトでは、システムは SAE プラグインによって以下のタイプのサービスに影響を与えるイベントを生成できます。

MPLS VPN のサービスに影響を与えるイベント
プロバイダー・エッジ (PE) またはカスタマー・エッジ (CE) のルーター上、またはディスカバーされたいずれかの MPLS VPN 内の CE ルーターを指しているいずれかの PE インターフェース上において、重大度 5 (重大) の障害イベントが発生したときに生成される合成イベント。 生成される SAE は、障害イベントが発生した MPLS VPN デバイス・コレクションを表す、ディスカバリー内の論理エンティティーのエンティティー ID に関連付けられます。

PE から CE へのインターフェースがメンバー・リストにすべて追加されます。このメンバー・リスト内のいずれかのインターフェースでイベントが発生すると、システムによって合成 MPLS VPN SAE が生成されます。

SAE イベントの生成は、ネットワーク・ディスカバリーの一環として BGPPeerNextHopInterface ディスカバリー・エージェントを有効にすることにより、コア・ネットワーク内のより深いインターフェース依存関係に基づいて有効にすることができます。 このエージェントは、AddLayer3VPNInterfaceDependency.stch スティッチャーを呼び出します。このスティッチャーは、VPN 内に存在するすべての PE からコア・プロバイダー・ルーター (P) へのインターフェース、および P から PE へのインターフェースを判別します。これらの PE -> P インターフェースおよび P ->PE インターフェースは、依存関係リストに追加されます。この依存関係リスト内のいずれかのインターフェースでイベントが発生すると、システムによって合成 MPLS VPN SAE が生成されます。メンバー・リスト内のいずれかのインターフェースにおけるイベントに基づいて、MPLS VPN SAE が既に生成されている場合、依存関係リスト内のインターフェースにおけるイベントはすべて、その既に生成されている MPLS VPN SAE の関連イベントとして追加されます。

BGPPeerNextHopInterface ディスカバリー・エージェントおよび AddLayer3VPNInterfaceDependency.stch スティッチャーについて詳しくは、*IBM Tivoli Network Manager IP Edition* ディスカバリー・ガイドを参照してください。

IP パスのサービスに影響を与えるイベント
ネットワーク・パス GUI を使用して作成された、いずれかの IP パス内にあるデバイス上でイベントが発生したときに生成される合成イベント。 生成される SAE は、イベントが発生したデバイスを含む IP パスに対応するエンティティー ID に関連付けられます。

カスタマイズ可能なサービスに影響を与えるイベント
SAE プラグインは、カスタム・サービスと関連付けられているデバイス上でイベントが発生したときに生成される合成イベントに対して構成することができます。

Fix Pack 3 この構成を実行するには、以下のタスクを実行する必要があります。

1. カスタム・サービスのデータを収集するように、ディスカバリー・エンジン `ncp_disco` を構成します。この構成を実行するには、カスタム・スティッチャーを記述してカスタム・サービスを定義し、そのスティッチャーが、関連する標準の Network Manager スティッチャーによって呼び出されるようにします。

   注**:** カスタム・スティッチャーの記述について不明な点がある場合は、IBM サポートにお問い合わせください。
2. NCIM キャッシュを更新して、カスタム・サービスのデータを `itnmService` NCIM データベース表に保管します。
3. `config.serviceTypes` SAE プラグイン・データベース表を更新して、新しいカスタム・サービスのデータを保管します。

Fix Pack 3 ncp_disco および itnmService NCIM データベース表の詳細については、「*IBM Tivoli Network Manager IP Edition* ディスカバリー・ガイド」を参照してください。

Fix Pack 3 複数のタイプのサービスに影響を与えるイベントを生成するように SAE プラグインを構成することができます。例えば、MPLS VPN エッジ・エンティティーの SAE イベント (あるタイプの SAE) と、MPLS VPN コア・エンティティーの SAE イベント (別のタイプの SAE) を作成するように、このプラグインを構成することができます。SAE プラグインの構成に使用した SAE プラグイン・データベース表を使用してください。

関連タスク:

197 ページの『SAE プラグインの構成』
ここでは、SAE プラグインの構成方法を説明します。

198 ページの『SAE プラグインへの SAE タイプの追加』
デフォルトで提供されている 3 つのタイプ以外の SAE を生成するように、SAE プラグインを構成することができます。例えば、MPLS VPN エッジ・エンティティーの SAE イベント (あるタイプの SAE) と、MPLS VPN コア・エンティティーの SAE イベント (別のタイプの SAE) を作成するように、このプラグインを構成することができます。

関連資料:

279 ページの『SAE プラグイン・データベース』
SAE プラグイン・データベース表は、MPLS VPN や IP パスなどのサービスに対して、SAE プラグインがサービスに影響を与えるイベントを生成できるようにします。

280 ページの『config.serviceTypes テーブル』
config.serviceTypes テーブルは、SAE プラグインの構成情報を保管します。

zNetView プラグイン

ここでは、プラグインの前提条件、およびプラグインに関連する構成の詳細について説明します。

必須フィールド

このプラグインでは、イベント・ゲートウェイによって渡されるイベントに、以下のフィールドが取り込まれていることが想定されています。

- Acknowledged

このプラグインでは、alerts.status テーブルに以下のカスタム・フィールドが存在している必要があります。

表 *38.* 適応ポーリング・プラグインのオプション構成

フィールド名	タイプ	説明
NmosClassName	64 文字のストリング	イベントの生成対象となったデバイスのクラス。この値は、NCIM トポロジー・データベースのシャーシ・テーブルから取得されます。

表 38. 適応ポーリング・プラグインのオプション構成 (続き)

フィールド名	タイプ	説明
NmosEntityType	32 ビット整数	イベントの生成対象であるエンティティーのタイプ。この値は NmosEntityId フィールドに指定されます。

プラグインの構成

ncmonitor.gwPluginConf テーブルで次のパラメーターを設定する必要があります。

表 39. 適応ポーリング・プラグインのオプション構成

パラメーター名	値	目的	デフォルト
StitcherSubDir	このプラグイン用のスティッチャーが含まれている、`$NCHOME/precision/eventGateway/stitchers/` ディレクトリー内のサブディレクトリーの名前。	このディレクトリーの名前を指定することで、zNetView プラグインのみがそのスティッチャーを解析できます。	`zNetView`

ncmonitor.gwPluginConf テーブルでは、オプションで以下のパラメーターを設定することができます。

表 40. 適応ポーリング・プラグインのオプション構成

パラメーター名	値	目的	デフォルト
StartupStitcher	初期化時に実行する、`$NCHOME/precision/eventGateway/stitchers/` 内のサブディレクトリーにあるスティッチャーの名前。	開始スティッチャー名が指定されている場合、開始時、SIGHUP 時、またはフェイルオーバー時かフェイルバック時に、このスティッチャーが引数なしで実行されます。	CheckAdditionalFields
SchemaFile	初期化時に解析する、`$NCHOME/etc/precision/` 内のスキーマ・ファイルの名前。	スキーマ・ファイル名が指定されている場合、開始スティッチャーが実行される前の、開始時、SIGHUP 時、またはフェイルオーバー時かフェイルバック時に、このファイルが解析されます。	なし

関連資料:

280 ページの『ncmonitor 内の ncp_g_event プラグイン・データベース表』
ここでは、ncmonitor データベース内にあるイベント・ゲートウェイ構成表と、各表に含まれている情報の種類について説明します。 これらの表のほとんどは、イベント・ゲートウェイのプラグイン構成に関係します。

プラグインのサブスクリプション

ここでは、各プラグインがサブスクライブするイベント・マップおよびイベント状態について説明します。

各イベント・ゲートウェイ・プラグインは、それぞれ異なるイベント・マップ・セットをサブスクライブします。

適応ポーリング・プラグイン

適応ポーリング・プラグインは、以下のイベント・マップをサブスクライブします。

1. EntityFailure
2. EntityIfDescr
3. EntityMibTrap
4. genericip-event
5. ItnmLinkDownIfIndex
6. ItnmMonitorEventNoRca
7. LinkDownIfDescr
8. LinkDownIfIndex
9. LinkDownIfName
10. NbrFail
11. NbrFailIfDescr
12. OSPFIfStateChange
13. OSPFIfStateChangeIP
14. PollFailure
15. PrecisionMonitorEvent

適応ポーリング・プラグインは、以下のイベント状態をサブスクライブします。

1. Cleared
2. Deleted
3. Occurred
4. ReAwakened
5. ReOccurred
6. ReSync
7. Updated

Disco プラグイン

Disco プラグインは以下の再構成イベント・マップをサブスクライブします。

フェイルオーバー・プラグイン

フェイルオーバー・プラグインはイベント・マップ ItnmHealthChk をサブスクライブします。また、以下のイベント状態をサブスクライブします。

1. Cleared

2. Occurred
3. ReAwakened
4. ReCleared
5. ReOccurred
6. Resolution
7. ReSync

RCA プラグイン

RCA プラグインは、以下のイベント・マップをサブスクライブします。

1. EntityFailure
2. EntityIfDescr
3. EntityMibTrap
4. ItnmLinkDownIfIndex
5. LinkDownIfDescr
6. LinkDownIfIndex
7. LinkDownIfName
8. NbrFail
9. NbrFailIfDescr
10. OSPFIfStateChange
11. OSPFIfStateChangeIP
12. PollFailure
13. PrecisionMonitorEvent

RCA プラグインは、以下のイベント状態をサブスクライブします。

1. Cleared
2. Deleted
3. Occurred
4. ReAwakened
5. ReOccurred
6. ReSync
7. Updated

SAE プラグイン

SAE プラグインは、以下のイベント状態をサブスクライブします。

MPLS VPN SAE プラグイン
このプラグインは、イベント状態「ReSync」をサブスクライブします。

IP パス・プラグイン
このプラグインは、イベント状態「ReSync」をサブスクライブします。

ITNM サービス・プラグイン
このプラグインは、イベント状態「ReSync」をサブスクライブします。

zNetView プラグイン

zNetView プラグインは、以下のイベント・マップをサブスクライブします。

1. EntityIfDescr
2. EntityFailure
3. EntityMibTrap
4. genericip-event
5. ItnmLinkDownIfIndex
6. ItnmMonitorEventNoRca
7. LinkDownIfIndex
8. LinkDownIfDescr
9. LinkDownIfName
10. NbrFailIfDescr
11. NbrFail
12. OspfIfState
13. OSPFIfStateChange
14. OSPFIfStateChangeIP
15. PollFailure
16. PrecisionMonitorEvent
17. Reconfiguration

zNetView プラグインは、以下のイベント状態をサブスクライブします。

1. Information
2. Occurred
3. ReAwakened
4. ReOccurred
5. Resolution
6. ReSync
7. Updated

根本原因の分析

ルート原因分析 (RCA) は、1 つ以上のデバイスのアラートの根本原因を判別するプロセスです。ルート原因分析 (RCA) プラグインはイベント・ゲートウェイから強化されたイベントのサブセットを受け取り、どのイベントが根本原因であり、どのイベントが副次であるかを判別します。 RCA は、ネットワークを介するトラフィックのルーティングに影響を与えるイベントのみを受信します。

ネットワーク上の障害状況においては、あるデバイスの障害条件がもとで他のデバイスがアクセス不能になることから、通常は複数のアラートが生成されます。生成されたアラートには、アクセス不能であるすべてのデバイスが示されます。
Network Manager はイベント情報とトポロジー情報を相関させることによって根本原因分析を行い、それによって他のネットワーク障害によって一時的にアクセス不能になっているデバイスを識別します。一時的にアクセス不能のデバイスについ

てのアラートは抑止され、オリジナルの根本原因のアラートの副次として表示されます。根本原因のアラートはアラート・リストとトポロジー・マップに表示されます。severity_from_causetype という ObjectServer の自動化が作成され、有効になっている場合、これらの根本原因アラートはオペレーターがこれを容易に識別できるように最高の重大度を持つものとして表示されます。

関連概念:

140 ページの『イベント・ゲートウェイ・プラグイン』
イベント・ゲートウェイ・プラグインは、イベント・ゲートウェイ・プロセスのモジュールであり、強化されたイベントをイベント・ゲートウェイから受け取って、そのイベントに対してさらなるイベント・エンリッチまたはその他のアクションを実行します。

関連資料:

140 ページの『プラグインの説明』
ここでは、各イベント・ゲートウェイ・プラグインの仕組みについて説明します。

RCA のクイック・リファレンス

ここでは、イベントが RCA プラグインを通過するときにどのように処理されるかを説明します。

RCA プラグインの目的は、イベント内のデータおよび RCA スティッチャー内でコーディングされているルールに基づいて判断を行い、他のイベントによって引き起こされるイベント、または他のイベントを引き起こすイベントを識別することです。以下の表に、手順の説明を示します。

表 *41. RCA* のクイック・リファレンス

アクション	その他の情報
1. イベント・ゲートウェイからイベントを受け取ります。 RCA プラグインは、イベント・マップおよびイベント状態のサブスクリプション要件をこのイベントが満たしていることを確認します。これは、RCA の用語で「トリガー・イベント」と呼ばれます。このイベントによって RCA プラグインのアクティビティーがトリガーされるためです。	152 ページの『プラグインのサブスクリプション』
2. イベントが RCA プラグインの mojo.events データベースに挿入されます。RCA スティッチャーはこのデータベースからイベントを取得して処理できます。	276 ページの『mojo.events イベント・データベース表』
3. イベントは ProcessEvent.stch に渡され、そこで RCA スティッチャーによる根本原因分析処理が行われます。	161 ページの『RCA スティッチャー』

関連タスク:

193 ページの『イベント・マップ・サブスクリプションの変更』
プラグインがサブスクライブしているイベント・マップを変更することができます。例えば、新しいイベント・マップを追加し、そのイベント・マップによって処理されるイベントに対してシステムによる RCA を実行する場合は、そのイベント・マップを RCA プラグインのサブスクリプション・リストに追加する必要があります。

優先順位値

イベントを処理するためのイベント・マップを選択すると同時に、数値の優先順位値がイベントに関連付けられます。この優先順位値は、同じエンティティーに対して複数のイベントがある場合に RCA プラグインによって使用されます。 エンティティーで最高の優先順位値を持つイベントは、他のイベントの抑止に使用されます。

優先順位値は、イベント・ゲートウェイの config.precedence テーブルを使用してイベント ID に対して構成します。 config.precedence テーブルは EventGatewaySchema.cfg 構成ファイル内で構成します。 このファイルの場所は、`$NCHOME/etc/precision/EventGatewaySchema.cfg`です。

次の例は、config.precedence テーブルを EventGatewaySchema.cfg 構成ファイル内で構成する方法を示しています。

優先順位値の構成に関連するコード・セクションを、この例の以下の行にリストします。 この例の insert は、EventId フィールドが SNMPTRAP-LinkDown に設定されたすべてのイベントに対して優先順位値 910 を割り当てるように、イベント・ゲートウェイを構成します。これらは Tivoli Netcool/OMNIbus プローブから発信されるトラップ・イベントです。 このコードには、config.precedence テーブルへの insert が含まれています。

次の表では、この例の関連する行について説明します。

表 42. 着信イベント・フィルターに関連するコードの行

行番号	説明
1	config.precedence テーブルへの insert を作成して、着信フィルターを構成します。
3	config.precedence テーブルの Precedence フィールド内への insert を指定します。
10	Precedence フィールドを値 910 に設定します。

```
1]    insert into config.precedence
2]    (
3]        Precedence,
4]        EventMapName,
5]        NcoEventId
6]    )
7]    values
8]    (
9]        910,
10]       "LinkDownIfIndex",
11]       "SNMPTRAP-LinkDown"
12]   );
```

関連資料:

271 ページの『config.precedence テーブル』
config.precedence テーブルは、イベントをイベント ID 別にリストしており、複数のイベントが同じインターフェース上で発生するときに、優先するイベントを判別するのに必要な情報が含まれています。また、config.precedence テーブルは、イベント ID に基づいて、Tivoli Netcool/OMNIbus からのイベントを処理するために

使用するイベント・マップを判別します。

デフォルトの優先順位値

慣例として、イベント・ゲートウェイは RCA プラグイン内において特殊な重要性を持つ事前定義のしきい値を割り当てます。

イベント・ゲートウェイの構成時に、独自の優先順位値を指定することができます。

以下のいずれかの条件を満たすイベントには、高い優先順位値を指定する必要があります。

- プロトコル・スタックの下位層にあるエンティティーのイベント。例えば、物理ポートで障害が発生したという確認には、そのインターフェース上の IP 層の問題より高い優先順位を指定します。
- 問題をより明確に示すイベント。例えば、ping 失敗イベントとリンクダウン・イベントを対比してみてください。より明確性の低いイベントは ping 失敗です。これは、ICMP パケットがインターフェースに到達できなかったことが原因で起こった可能性があります。言い換えれば、これは、ポーリング・ステーションとインターフェース間のネットワーク問題が原因と考えられます。より明確性の高いイベントは、リンクがダウンしたことを明示的に示す SNMP トラップです。こちらの方が、インターフェース自体またはそこに直接接続されている隣接デバイスの問題をより積極的に確認しているためです。

次の表に、イベント・ゲートウェイがデフォルトで割り当てる優先順位値をリストします。

表 *43.* デフォルトの優先順位値

値	意味	イベント例
0	他の問題の原因となり得ないイベントに割り当てられます。 RCA の実行中、このイベントは他のイベントを抑止できませんが、自身を抑止することはできます。	SYSLOG-cisco-ios-SYS-CPUHOG SYSLOG-cisco-ios-BGP-NOTIFICATION
300	デバイス上の障害を示唆する (ただし、必ずしも明確に示しているわけではない)、正式ではないイベント用に予約されています。例えば、デバイスへの到達の失敗は、必ずしもそのデバイスに問題があることを示すわけではありません。この障害はポーリング・ステーションとデバイス間の問題によって引き起こされる可能性があります。	probeping-icmptimeout SNMPTRAP-IETF-OSPF-TRAP-MIB-ospfIfStateChange
600	プロトコル障害を対象としています。プロトコル・スタックの下位層で識別された障害には、高い優先順位が割り当てられるはずです。例えば、OSPF は IP より上部で実行されるため、OSPF の障害は IP の障害より優先順位が低くなることが予想されます。	SNMPTRAP-IETF-OSPF-TRAP-MIB-ospfIfConfigError

表 43. デフォルトの優先順位値 (続き)

値	意味	イベント例
900	リンクダウンまたは ping 失敗 (および他のほとんどのイベント) を間接的に意味する確認済みの物理障害に割り当てられます。	SNMPTRAP-cisco-CISCO-WIRELESS-IF-MIB-cwrTrapLinkQuality
910	リンクダウンまたは ping 失敗を直接的に示す確認済みの物理障害に割り当てられます。	SNMPTRAP-linkDown SYSLOG-smc-switch-linkDown
10 000	他の問題が原因で起こることがあり得ないイベントに割り当てられます。 RCA の実行中、このイベントは他のイベントによって抑止されることはあり得ませんが、根本原因となって、他のイベントを抑止することはあります。	SYSLOG-cisco-ios-CI-SHUTDOWN SNMPTRAP-riverstone-RIVERSTONE-NOTIFICATIONS-MIB-rsEnvirHotSwapOut

ポーリング・プログラム・エンティティー

ここでは、ポーリング・プログラム・エンティティーとは何であるか、およびその構成方法を説明します。

ポーリング・プログラム・エンティティーはポーリング・ステーションとも呼ばれるサーバーのことで、このサーバーから Network Manager がデバイスをポーリングします。ポーリング・ステーション (通常は Network Manager サーバー) がご使用のネットワーク・ドメインのスコープ内にない場合、RCA プラグインが分離抑止を実行できるように、入口インターフェースの IP アドレスまたは DNS 名をポーリング・プログラム・エンティティーとして指定する必要があります。これは、ポーリング・ステーションとの間でネットワーク・パケットの送受信が行われるディスカバリー・スコープ内のインターフェースです。

ポーリング・プログラム・エンティティーは、ローカル Network Manager サーバーを表すときに使用するサーバー名で、NcpServerEntity フィールド内の config.defaults テーブルに格納されています。 Network Manager サーバーがディスカバリーのスコープ内にない場合、NcpServerEntity フィールドに値が必要です。

NcpServerEntity フィールドは次のように構成する必要があります。

表 44. *NcpServerEntity* フィールドの構成設定

Network Manager サーバーがディスカバリー・スコープ内にあるかどうか	**NcpServerEntity** フィールドの値
はい	空ストリング
いいえ	入口インターフェースの IP アドレスまたは DNS 名

次の図は、Network Manager サーバーがディスカバリー・スコープの外部にある場合の入口インターフェース (円で囲んだ部分) を示しています。

注: イベント・ゲートウェイが NCIM データベース内でポーリング・プログラム・エンティティーを見つけるためには、ディスカバリーを少なくとも 1 回実行している必要があります。

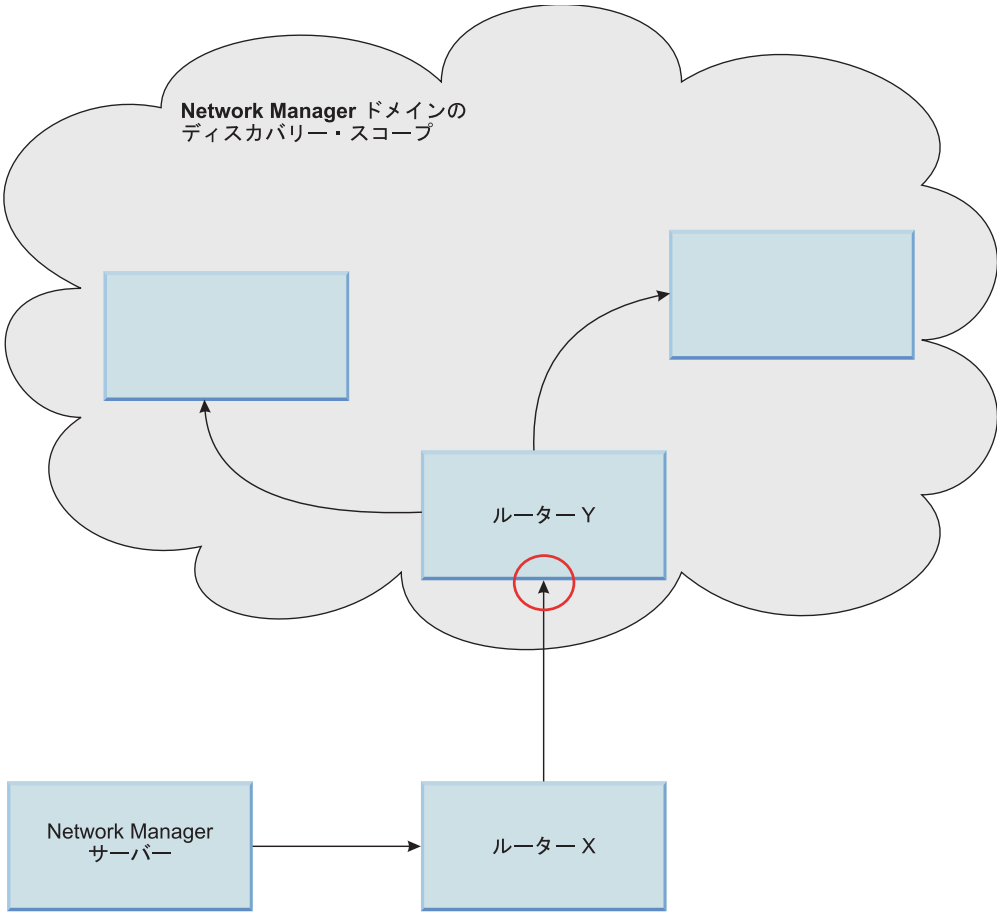


図 6. 入口インターフェース

関連タスク:

201 ページの『ポーリング・プログラム・エンティティーの構成』

Network Manager サーバーがご使用のネットワーク・ドメインのスコープ内にない場合に RCA プラグインが分離抑止を実行できるようにするには、入口インターフェースの IP アドレスまたは DNS 名をポーリング・プログラム・エンティティーとして指定します。

RCA および非管理状況

ここでは、RCA プラグインが非管理状態 (保守状態とも言います) にあるデバイスからのイベントを処理する方法について説明します。

RCA プラグインが非管理状態のデバイスからのイベントを処理するときに使用する方法は、RCA プラグイン構成ファイル `$NCHOME/etc/precision/RCASchema.cfg` の `HonourManagedStatus` に設定されている値によって制御されます。このフィールドは、以下のいずれかの値になります。

- 1 (デフォルト値): 管理状況のイベントを受け取るように RCA プラグインに指示します。非管理のデバイスからのイベントはすべて無視されます。

- 0: 非管理のデバイスからのイベントを通常のイベントとして処理するように RCA プラグインに指示します。

RCA プラグインはそれぞれの場合に、イベントの NmosManagedStatus フィールドを調べて、デバイスが非管理であるかどうかを判別します。

RCA プラグインが管理状況のイベントを受け取る (HonourManagedStatus = 1) ように構成されていると想定すると、非管理のデバイスからのイベントは根本原因にはなり得ず、それを抑止することはできません。

イベントのイベント状態が「ReOccurred」であり、この同じイベントが以前に発生したときはデバイスが管理状態であったことを示していた場合、mojo.events データベース内のイベント・レコードは更新され、その NmosCauseType、NmosSerial、および SuppressionState は 0 にリセットされ、それによって事実上、このイベントを無視するように RCA プラグインに指示します。再発生イベントの管理状況は、管理状況がインターフェースなどのエンティティーのプロパティーであるために変更される場合があります。エンティティーの管理状況は、エンティティー・レコード内のフィールドに保管されます。さらに、エンティティーで発生したイベントには、NmosManagedStatus というフィールドも含まれており、このフィールドに、イベントが発生した時点における エンティティーの管理状況が記録されます。そのため、エンティティーが管理されているときにイベントが発生した場合でも、その後、エンティティーが非管理状態のとき (すなわち、エンティティーが管理状態から非管理状態に変更された後) に同じイベントが再発生する可能性があります。

以下のシナリオでは、イベントの管理状況が、そのイベントの以後の発生時に変更されている場合に、RCA プラグインがそのイベントを処理する方法について説明します。

イベントが管理状況から非管理状況に変更された場合

順序を以下に示します。

1. 最初に発生したイベント (例えば、ping 失敗イベントなど) は RCA で通常のイベントとして処理されます。そのイベントでは NmosManagedStatus が 0 に設定されているためです。0 は、イベントが最初に発生した時点ではエンティティーが管理状況だったことを示しています。
2. その後、しばらくたってから、インターフェース・エンティティーが非管理状況に設定されます。つまり、インターフェースの ManagedStatus 値が 1 になります。
3. そのインターフェースでイベントが再発生します。
4. 再発生した ping 失敗イベントにはフィールド値 NmosManagedStatus = 1 が含まれるようになりますが、このイベントが以前に発生したときのフィールド値 NmosManagedStatus = 0 は引き続きデータベース mojo.events に残っています。
5. RCA プラグインは、「ReOccurred」(または「Updated」) イベントに関して、フィールド NmosManagedStatus の値が 0 から 1 に変更されたことを検出します。

6. RCA プラグインはデータベース mojo.events 内のイベント・レコードを更新し、それ以降、そのイベントを削除済みイベントを処理する場合と同様に処理します。つまり、このイベントは他のイベントを抑止できなくなったため、抑止されていたイベントをすべて再処理します。

イベントが非管理状況から管理状況に変更された場合

順序を以下に示します。

1. 最初に発生したイベント (例えば、ping 失敗イベントなど) が、NmosManagedStatus が 1 (イベントが最初に発生した時点でエンティティーが非管理状況だったことを示す) に設定された状態で到着します。そのため、イベントは削除済みイベントと同様に処理され、他のイベントを抑止することはできません。
2. その後、しばらくたってから、インターフェース・エンティティーが管理状況に設定されます。つまり、インターフェースの ManagedStatus 値が 0 になります。
3. そのインターフェースでイベントが再発生します。
4. 再発生した ping 失敗イベントにはフィールド値 NmosManagedStatus = 0 が含まれるようになりますが、このイベントが以前に発生したときのフィールド値 NmosManagedStatus = 1 は引き続きデータベース mojo.events に残っています。
5. RCA プラグインは、「ReOccurred」(または「Updated」) イベントに関して、フィールド NmosManagedStatus の値が 1 から 0 に変更されたことを検出します。
6. RCA プラグインはデータベース mojo.events 内のイベント・レコードを更新し、それ以降、そのイベントを通常のイベントとして処理します。このイベントは、他のイベントを抑止できるようになります。

関連資料:

278 ページの『config.defaults データベース表』
config.defaults データベース表は、RCA プラグイン・イベント・キューの構成データを保管します。

RCA スティッチャー

RCA スティッチャーは、トリガー・イベントが RCA プラグインを通過する間にこれを処理します。各 RCA スティッチャーはそれぞれ特定のタスクを実行します。1 つのスティッチャーが RCA プラグインによって呼び出されると、残りのスティッチャーは特定の順序で実行されます。

RCA プラグイン・スティッチャーは $NCHOME/precision/eventGateway/stitchers/RCA にあります。

スティッチャー言語については、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。

RCA スティッチャーのシーケンス

ここでは、RCA プラグインによって呼び出されるスティッチャーについて、および根本原因を判別するためにスティッチャーが実行されるシーケンスについて説明します。

1. イベント・ゲートウェイが RCA プラグインにイベントを渡すと、そのイベントを処理するために ProcessEvent.stch スティッチャーが呼び出されます。このイベントはトリガー・イベントと呼ばれます。
2. ProcessEvent.stch スティッチャーは、以下の表に示すように、トリガー・イベントのイベント状態に応じて、呼び出すスティッチャーを決定します。

トリガー・イベントのイベント状態	呼び出されるスティッチャー
Occurred ReAwakened ReOccurred Resync Updated	ProcessProblemEvent.stch
Cleared Deleted	ProcessResolutionEvent.stch イベントがクリアまたは削除されると、RCA プラグインは、クリアまたは削除されたイベントによって抑止されていたすべてのイベントを再処理します。

3. トリガー・イベントの抑止を試みるために、2 つのトリガーが ProcessProblemEvent.stch スティッチャーによって呼び出されます。これらのトリガーは以下のとおりです。
   - SuppressTrigger.stch スティッチャーは、既存のイベントによってトリガー・イベントを抑止できるかどうかを判別します。
   - OSPF または BGP イベントの場合、PeerEntitySuppression.stch スティッチャーは、ピア・エンティティーの他のイベントによってトリガー・イベントを抑止できるかどうかを判別します。

   SuppressTrigger.stch または PeerEntitySuppression.stch スティッチャーがトリガー・イベントを抑止できない場合は、トリガー・イベントが根本原因になります。
4. RCA プラグインは、同じエンティティーの他のイベントによってトリガー・イベントが抑止されるかどうかを検査します。つまり、検査では、トリガー・イベントがエンティティーのマスター・イベントでないかどうかが評価されます。イベント抑止を実行するのは、エンティティーのマスター・イベントです。そのため、トリガー・イベントがマスター・イベントでない場合、トリガー・イベントは他のイベントを抑止できません。後続のステップでは、ProcessProblemEvent.stch が他のスティッチャーを呼び出し、それらのスティッチャーがトリガー・イベントを使用して他のイベントの抑止を試みます。各スティッチャーは順に実行されます。
5. EntitySuppression.stch スティッチャーは、トリガー・イベントを使用して、子エンティティーの原則に基づき他のイベントを抑止します。エンティティーで最高の優先順位を持つイベントは、そのエンティティーの他のすべてのイベントを抑止します。

6. ContainedEntitySuppression.stch スティッチャーは、トリガー・イベントを使用して、子エンティティーの原則に基づき他のイベントの抑止を試みます。親エンティティーのイベントは、すべての子エンティティーのイベントを抑止します。
7. IsolatedEntitySuppression.stch スティッチャーは、トリガー・イベントを使用して、ダウンストリーム・エンティティーの原則に基づき他のイベントの抑止を試みます。
8. ConnectedEntitySuppression.stch スティッチャーは、トリガー・イベントを使用して、接続されているエンティティーの原則に基づき他のイベントの抑止を試みます。例えば、2 つのインターフェースが接続されていて、その両方のインターフェースにイベントが存在する場合、一方のインターフェースのイベントが他方のインターフェースのイベントを抑止します。

関連概念:

110 ページの『イベント状態』
イベント・ゲートウェイは、イベントのタイプ、およびイベント内の重大度フィールドと合計フィールドに基づいて、イベントに状態を割り当てます。 イベント状態は、イベント・プラグインによってイベントのサブスクライブ時に使用されるパラメーターの 1 つです。

RCA スティッチャーの定義済み定数

Network Manager は、RCA 抑止タイプと RCA 原因タイプの定義済み定数を提供します。新しい RCA スティッチャーをコーディングしたり、既存の RCA スティッチャーを変更したりする際、これらの定義済み定数を整数の変数に保管できます。

抑止タイプの定義済み定数は次のとおりです。

$RCA_NO_SUPPRESSION
抑止されません。根本原因イベントはこの状態になります。

$RCA_ENTITY_SUPPRESSION
同じエンティティーの他のイベントによって抑止されます。

$RCA_CONTAINED_SUPPRESSION
包含抑止。例えば、シャーシ・デバイス内に含まれているインターフェースの障害は、そのシャーシ・デバイスの障害によって抑止されます。

$RCA_ISOLATED_SUPPRESSION
分離抑止。単一のシャーシ・デバイスの下流で分離されているデバイスの障害は、分離しているシャーシ・デバイスの障害によって抑止されます。

$RCA_CONNECTED_SUPPRESSION
接続されているエンティティーのイベントによって抑止されます。

$RCA_PEER_SUPPRESSION
ピア・エンティティー抑止。

原因タイプの定義済み定数は次のとおりです。

$RCA_UNKNOWN_CAUSE
イベントの原因が不明です。

$RCA_ROOT_CAUSE
根本原因イベント。

$RCA_SUPPRESSED
抑止イベント。

注**:** スティッチャー・コードを使用して causeType 変数を $RCA_SUPPRESSED に設定することは決してしないでください。この設定は、基となる RCA コードによってのみ実行される必要があります。

RCA スティッチャーの説明

ここでは、各 RCA スティッチャーの仕組みについて説明します。

次の表に RCA スティッチャーを示します。

表 *45. RCA* スティッチャー

スティッチャー	説明
ConnectedEntitySuppression.stch	トリガー・イベントを使用して、接続されているエンティティーの原則に基づき他のイベントの抑止を試みます。例えば、2 つのインターフェースが接続されていて、その両方のインターフェースにイベントが存在する場合、一方のインターフェースのイベントが他方のインターフェースのイベントを抑止します。
ContainedEntitySuppression.stch	トリガー・イベントを使用して、子エンティティーの原則に基づき他のイベントの抑止を試みます。親エンティティーのイベントは、すべての子エンティティーのイベントを抑止します。
EntitySuppression.stch	トリガー・イベントを使用して、同一エンティティーの抑止の原則に基づき他のイベントの抑止を試みます。同一エンティティーで最高の優先順位を持つイベントは、そのエンティティーの他のイベントを抑止します。
IsolatedEntitySuppression.stch	トリガー・イベントを使用して、ダウンストリーム・エンティティーの原則に基づき他のイベントの抑止を試みます。
PeerEntitySuppression.stch	OSPF または BGP イベントの場合、既存の OSPF または BGP イベントによってトリガー・イベントを抑止できるかどうかを決定します。
ProcessEvent.stch	これはヘッド・スティッチャーです。トリガー・イベントが RCA プラグインに渡されるたびに、これが呼び出されます。ProcessEvent.stch スティッチャーは、トリガー・イベントのイベント状態に基づいて、呼び出すスティッチャーを決定します。 • ProcessProblemEvent.stch は、イベント状態が「Occurred」、「ReAwakened」、「ReOccurred」、「Resync」、および「Updated」のイベントを処理するために呼び出されます。 • ProcessResolutionEvent は、イベント状態が「Cleared」および「Deleted」のイベントを処理するために呼び出されます。

表 45. RCA スティッチャー (続き)

スティッチャー	説明
ProcessProblemEvent.stch	問題イベント (すなわち、イベント状態が「Occurred」、「ReAwakened」、 「ReOccurred」、「Resync」、 および「Updated」のイベント) を処理します。このスティッチャーは SuppressTrigger スティッチャーおよび PeerEntitySuppression を呼び出し、他のイベントを使用してトリガー・イベントの抑止を試みます。その後に、EntitySuppression、 ContainedEntitySuppression、ConnectedEntitySuppression、および IsolatedEntitySuppression スティッチャーをこの順序で呼び出して、トリガー・イベントを使用して他のイベントの抑止を試みます。
ProcessResolutionEvent.stch	解決イベント (すなわち、イベント状態が「Cleared」および「Deleted」のイベント) を処理します。
SuppressTrigger.stch	既存のイベントによってトリガー・イベントを抑止できるかどうかを決定します。
TimedEventSuppression.stch	このスティッチャーの目的は、RCA プラグインがフラップ・イベントを処理しないようにして、リソースを節約することです。 フラップできるイベントは、`EventCanFlap = 1` が設定されてイベント・ゲートウェイから渡されます。これらのイベントは、`TimedEscalation = 1` が指定されて mojo.events データベースに入れられ、そのまま 30 秒間置かれます。30 秒を過ぎると、TimedEventSuppression RCA スティッチャーは、30 秒以上経過し、なおかつ `TimedEscalation = 1` が設定されたイベントをすべて処理します。 **注:** イベントの処理を 30 秒待つことで、イベントを生成したエンティティーが安定し、フラップしないようになります。フラップするエンティティーの一例として、リンクダウン・イベントおよびリンクアップ・イベントを連続して次々と生成するインターフェースがあります (このインターフェースはこれらのイベントを 2 秒ごとに生成することもあります)。リンクアップ・イベントが RCA プラグインを通過する間に、ProcessResolutionEvent スティッチャーがリンクダウン・イベントを削除します。その結果、フラップ・イベントは 30 秒の待機時間中に削除されてしまっているため、TimedEventSuppression によって処理されることはありません。 処理の後、`TimedEscalation = 1` が設定されたすべてのイベントの TimedEscalation フィールドは 2 に設定され、それ以上の処理は行われないようになります。

関連概念:

110 ページの『イベント状態』
イベント・ゲートウェイは、イベントのタイプ、およびイベント内の重大度フィールドと合計フィールドに基づいて、イベントに状態を割り当てます。 イベント状態は、イベント・プラグインによってイベントのサブスクライブ時に使用されるパラ

メーターの 1 つです。

根本原因分析の例

これらの例では、さまざまなタイプのネットワーク・デバイスおよびインターフェースの考慮事項に基づいて、RCA プロセスが根本原因分析を実行する方法を示します。この例は実例となるように提供されているだけで、RCA が使用する原則のみ示すように意図されています。大規模ネットワークでの RCA はもっと複雑です。

図に示す色は、アクティブ・イベント・リスト内の以下のイベントの色と一致しています。

- 赤: 根本原因イベント。
- 紫: 副次 (抑止された) イベント。

アクティブ・イベント・リストでの根本原因イベントの識別と調査についての詳細は、「*IBM Tivoli Network Manager IP Edition* ネットワーク・トラブル・シューティング・ガイド」を参照してください。

RCA でのダウンストリームとアップストリームの定義

ここでは、RCA プラグインにおいてダウンストリームおよびアップストリームという用語がどのように適用されるかを説明します。

用語の定義

ダウンストリームおよびアップストリームという用語は、ポーリング・プログラム・エンティティーに関連して使用されます。

ダウンストリーム
: ポーリング・ステーションからトポロジー的に離れているが、2 番目のロケーションと同じ物理パスにあるネットワークのロケーションを示します。

アップストリーム
: ポーリング・ステーションからトポロジー的に近いが、2 番目のロケーションと同じ物理パスにあるネットワークのロケーションを示します。

複雑なネットワークでは、デバイスが非アクティブ化されると、ポーリング・ステーションからのデバイスの距離が変わります。この距離の変化が、デバイスがアップストリームかダウンストリームかということに影響を与えます。

例

以下の図は、アップストリーム・ロケーションとダウンストリーム・ロケーションの例を示しています。この例では、デバイス B はデバイス A のダウンストリームであり、デバイス A はデバイス B のアップストリームです。

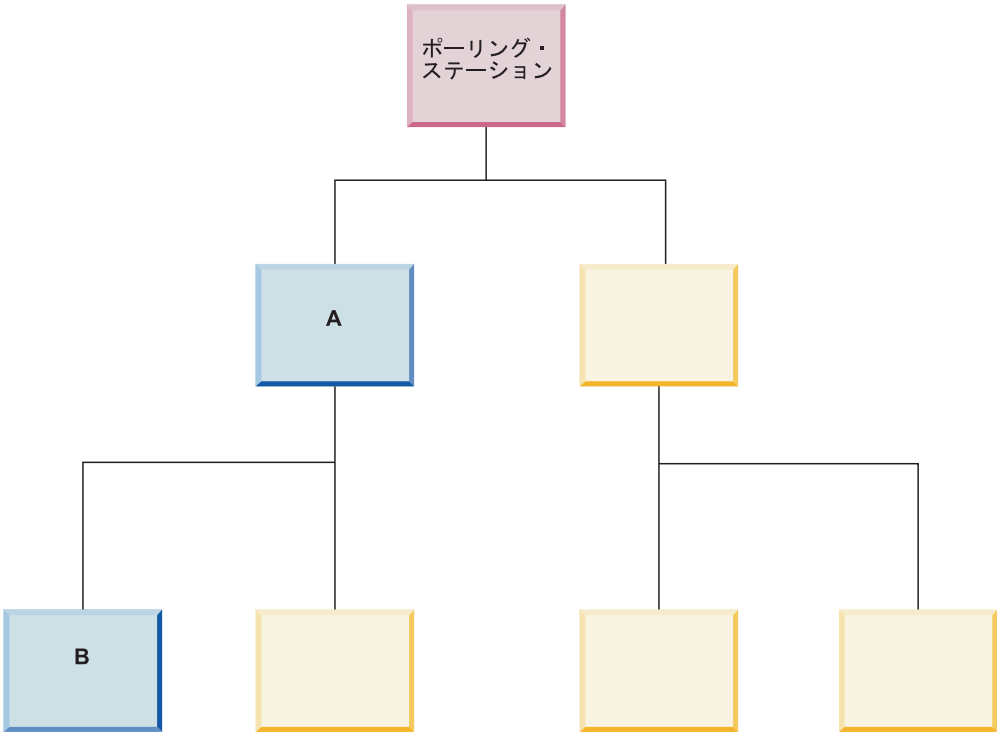


図 7. ダウンストリーム・デバイスおよびアップストリーム・デバイス

関連資料:

171 ページの『シャーシ・デバイスの分離抑止』
シャーシ・デバイス上の障害は、障害が発生したシャーシ・デバイスによって分離されるすべてのシャーシ・デバイス上の障害を抑止します。これは分離抑止 の例です。

174 ページの『ネットワークのエッジでのデバイスの分離抑止』
他のエンティティーとネットワーク間の単独接続である、論理インターフェースまたは物理インターフェース上の障害は、ダウンストリーム・エンティティーでの障害を抑止します。これは分離抑止 の例です。

シャーシ・デバイスおよびループバック・インターフェース

大抵の場合、RCA プロセスは、シャーシに障害が起こる場合、他の障害の根本原因はシャーシから始まっていると想定します。シャーシの障害は、組み込まれているインターフェース、接続されているインターフェース、およびダウンストリーム・シャーシ・デバイス上の障害を抑止します。

ループバック・インターフェースには、シャーシ・デバイス内に特殊な機能 (ルーターまたはスイッチ) があります。ループバック・インターフェースには、常にシャーシ・デバイスの IP アドレスと一致する IP アドレスが割り当てられます。。Network Manager IP Edition は、ディスカバリー時にループバック・インターフェースをシャーシに関連付けます。ループバック・インターフェースはシャーシ全体を表し、個別にポーリングすることができます。ループバック・インターフェース上の障害は、シャーシ・デバイス上の障害と全く同じようにして、接続されているエンティティーおよび組み込まれているエンティティー上の障害を抑止します。

シャーシ・デバイス、インターフェース、モジュール、およびカード上のイベントだけが、他のイベントを接続抑止できます。ただし、シャーシが他のシャーシ (またはドーター・カード) を接続抑止することはありません。

組み込まれているインターフェース:

シャーシの障害は、そのシャーシに組み込まれているインターフェース上のすべての障害を抑止します。

以下の図では、シャーシ・デバイス A 上の障害が、インターフェース b、c、および d 上の障害を抑止します。インターフェース b、c、d はすべてシャーシ・デバイス A 内に含まれます。

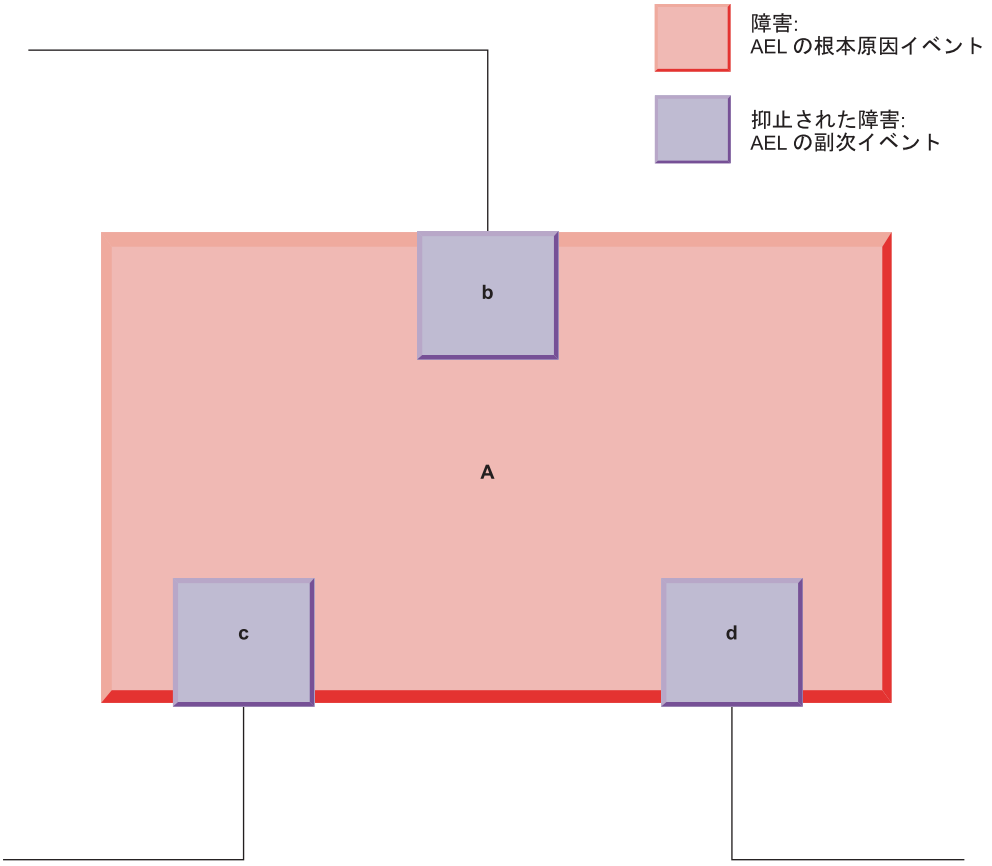


図 *8.* シャーシの障害は、組み込まれているインターフェース上の障害を抑止する

接続されているインターフェース:

シャーシの障害は、そのシャーシ・デバイスに接続されているインターフェース上のすべての障害を抑止します。障害は、アップストリーム・インターフェースとダウンストリーム・インターフェースの両方で抑止されます (ただし、それらのインターフェースが分離ポイントでない場合)。

以下の図では、デバイス A がインターフェース b、c、および d 上の障害を抑止しています。

注: インターフェースがグラフの分離ポイントである場合は、隣接するエンティティー上のイベントによってそのインターフェースを接続抑止することはできません。

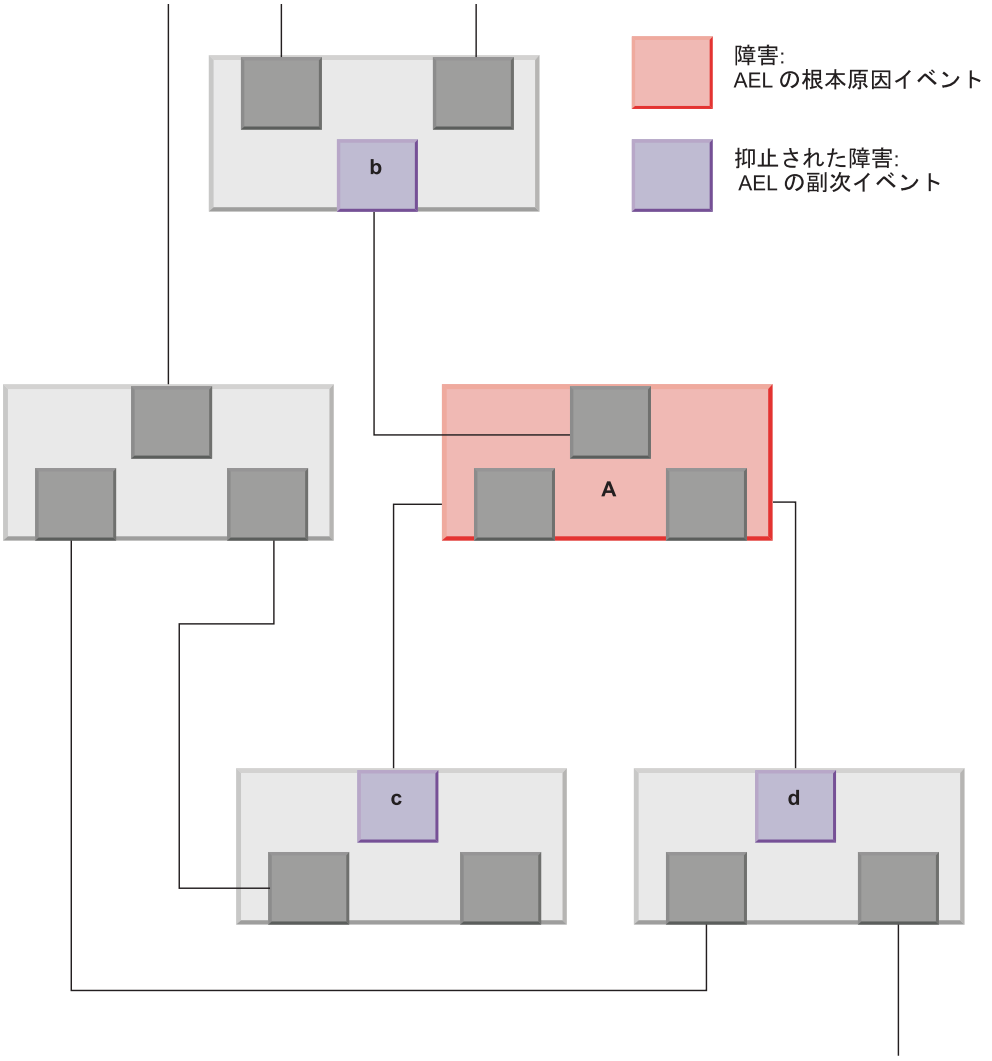


図 9. シャーシの障害は、接続されているインターフェース上の障害を抑止する

組み込まれているエンティティーに接続されるエンティティー:

シャーシ・デバイスには 1 つ以上のエンティティーが含まれることがあります。シャーシ・デバイス内に含めることができるエンティティーの例として、VLAN、カード、および仮想ルーターがあります。カードなどの組み込まれているエンティティーには 1 つ以上のインターフェースがある場合があります。

シャーシ・デバイス上の障害は、そのシャーシ・デバイス内に含まれる、いずれかのエンティティーに直接接続されたエンティティー上の障害を抑止します。以下の図では、エンティティー B はシャーシ・デバイス A 内に含まれています。シャーシ・デバイス A 上の障害は、デバイス D のインターフェース d 上およびデバイス E 上のインターフェース e 上の障害を抑止します。インターフェース d も e もエンティティー B に直接接続されます。

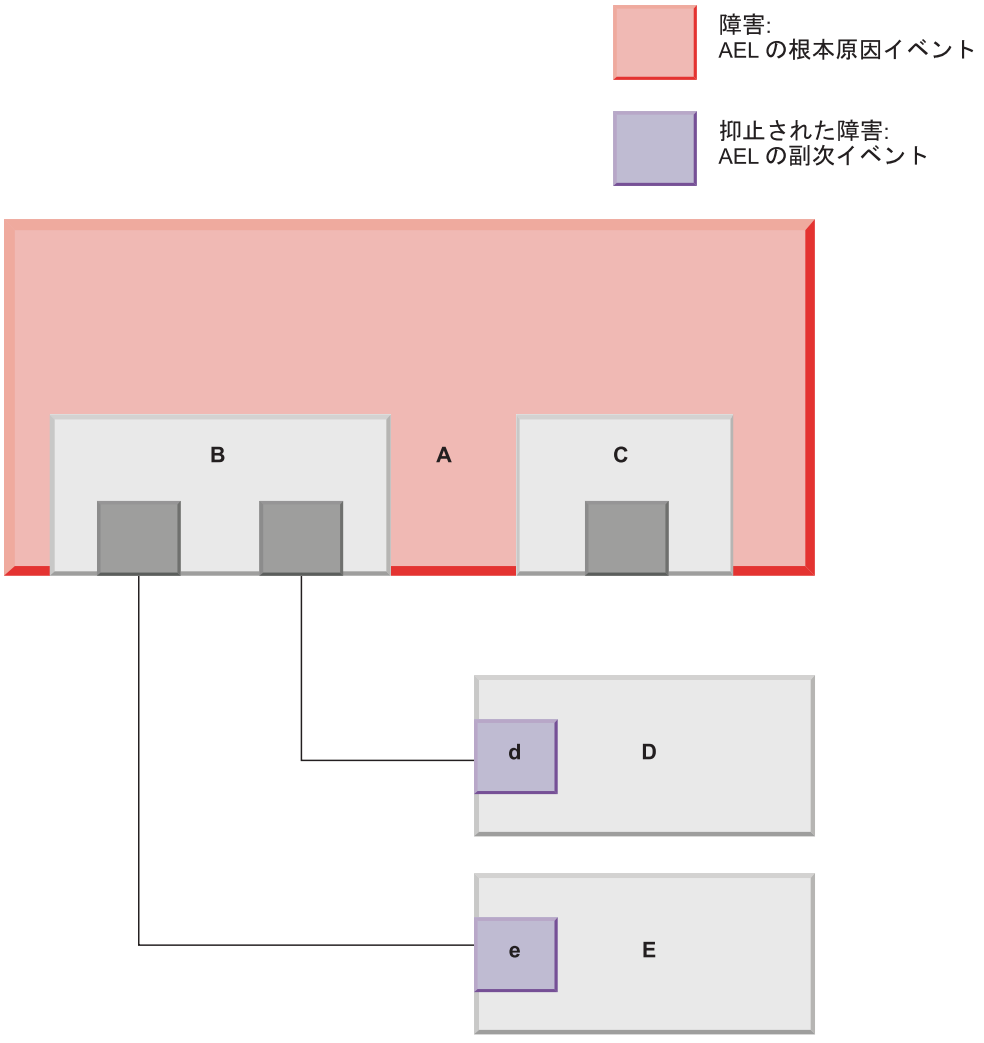


図 *10.* シャーシの障害は、組み込まれているエンティティーに接続されたデバイス上の障害を抑止する

シャーシ・デバイスの分離抑止:

シャーシ・デバイス上の障害は、障害が発生したシャーシ・デバイスによって分離されるすべてのシャーシ・デバイス上の障害を抑止します。これは分離抑止 の例です。

以下の図では、シャーシ・デバイス A 上の障害が、シャーシ・デバイス B、C、D 上の障害を抑止しています。シャーシ・デバイス B、C、D は、いずれもシャーシ・デバイス A によって分離されています。

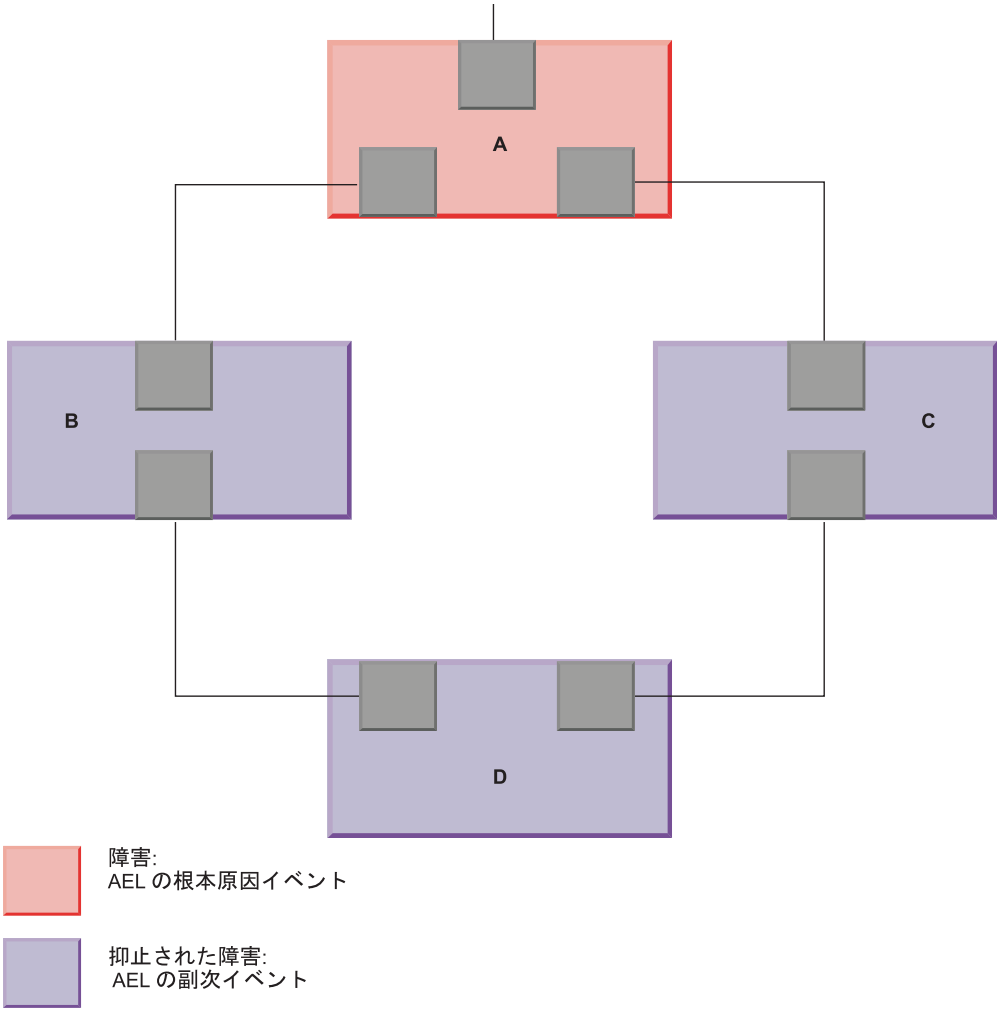


図 *11.* シャーシの障害は、ダウンストリーム・エンティティー上の障害を抑止する

関連資料:

166 ページの『RCA でのダウンストリームとアップストリームの定義』
ここでは、RCA プラグインにおいてダウンストリームおよびアップストリームという用語がどのように適用されるかを説明します。

Interfaces

インターフェースがダウンストリームの障害を分離している場合は、そのインターフェースの障害がダウンストリームの障害を抑止できます。

2 つのインターフェースが直接接続されている場合、標準インターフェースの障害は 2 つ目の物理インターフェースの障害を抑止できます。抑止ルールが最初に起動したインターフェースが、それ以外のインターフェースを抑止します。 1 つ目のインターフェースの障害が 2 つ目のインターフェースの障害によって抑止されるのは、インターフェースの障害がシャーシの障害によってまだ抑止されていない場合に限られます。

注: インターフェースがグラフの分離ポイントである場合は、隣接するエンティティー上のイベントによってそのインターフェースを接続抑止することはできません。

物理インターフェースには複数の論理インターフェースを含めることができます。物理インターフェース上の障害は、それに関連した論理インターフェース上の障害を抑止できます。論理インターフェースと外部隣接インターフェースの間に接続がある場合でも、物理インターフェースはそれに関連した論理インターフェースを抑止できます。抑止された物理インターフェース上のイベントでも、それに関連した論理インターフェース上のイベントを包含抑止できます。

一般には、シャーシ、インターフェース、モジュール、およびカード上のイベントだけが、他のイベントを接続抑止できます。

直接接続されたインターフェース:

2 つのインターフェースが直接接続されている場合、標準物理インターフェースの障害は 2 つ目の物理インターフェースの障害を抑止します。

直接接続されているインターフェースの抑止には、以下の制約が関連します。

- 包含抑止されているインターフェースは、抑止された状態で接続できません。
- 抑止されたインターフェースは接続されたインターフェースを抑止できます。

以下の図では、インターフェース a 上の障害が、直接接続されているインターフェース b 上のもっと新しい障害を抑止します。

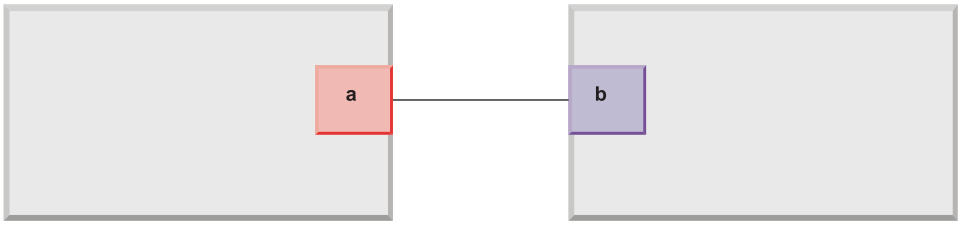


図 *12.* インターフェースの障害は、直接接続された隣接インターフェース上のもっと新しい障害を抑止する

関連した論理インターフェース**:**

物理インターフェース上の障害は、関連した論理インターフェース上の障害を抑止します。

以下の図では、物理インターフェース上の障害が、包含される論理インターフェース b と c 上の障害を抑止します。

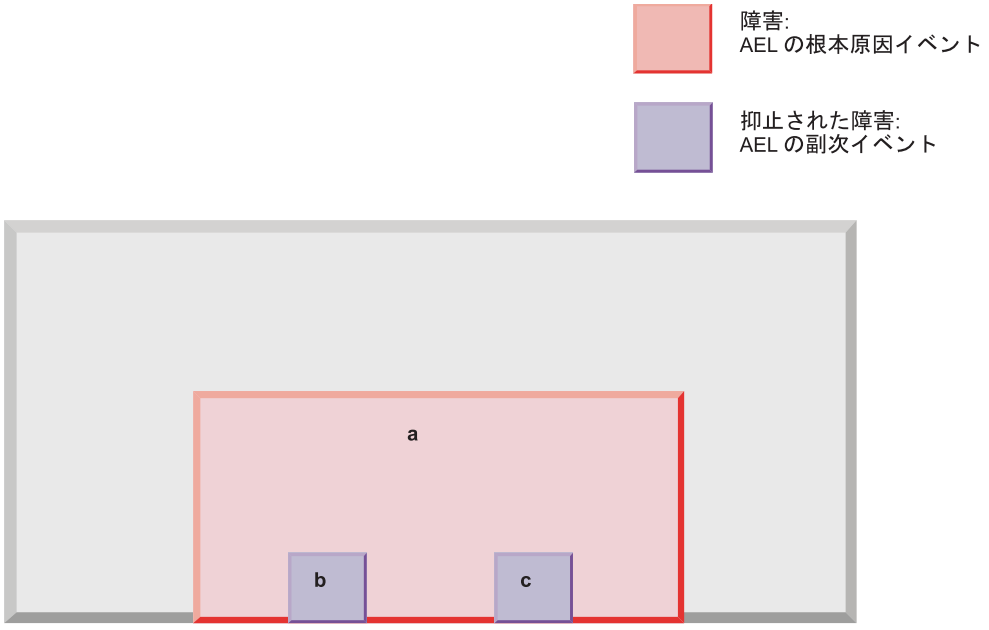


図 *13.* 物理インターフェースの障害は、包含される論理インターフェース上の障害を抑止する

ネットワークのエッジでのデバイスの分離抑止:

他のエンティティーとネットワーク間の単独接続である、論理インターフェースまたは物理インターフェース上の障害は、ダウンストリーム・エンティティーでの障害を抑止します。これは分離抑止 の例です。

以下の図では、デバイス A のインターフェース d 上の障害が、デバイス B、C、D とそれらのインターフェース上の障害を抑止しています。

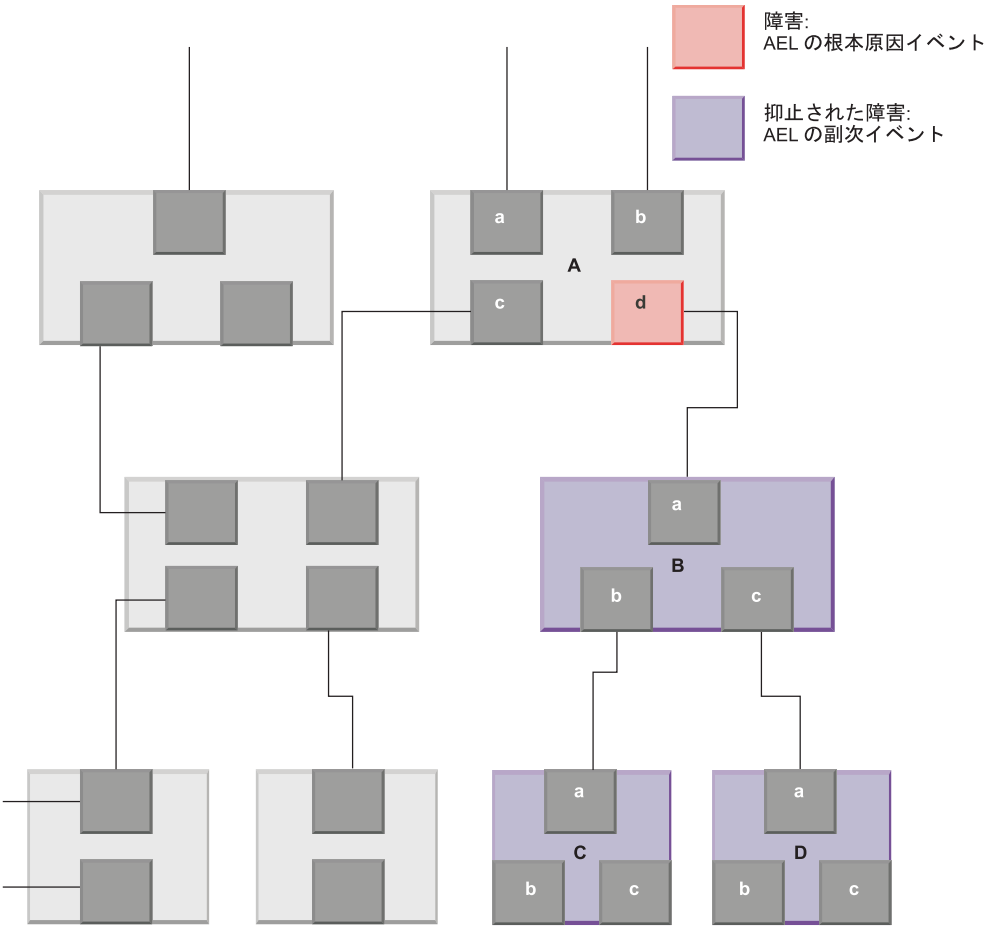


図 *14.* インターフェースの障害は、直接接続された隣接インターフェース上のもっと新しい障害を抑止する

関連資料:

166 ページの『RCA でのダウンストリームとアップストリームの定義』
ここでは、RCA プラグインにおいてダウンストリームおよびアップストリームという用語がどのように適用されるかを説明します。

RCA によって使用されるトポロジー・パスの検査

ネットワーク・デバイス間のトポロジー・パスをトポロジー相関の目的に使用できるかどうかを検査するには、RCA パス・ツールを使用します。

RCA パス・ツールについて

RCA パス・ツールは、分離抑止を伴う根本原因分析の状況のためのデバッグ支援機能を提供します。

RCA パス・ツールによって、A から Z への最短パスが分かります。ここで、A および Z はトポロジーの中の 2 つのノード (例えば、デバイス) です。RCA パス・ツールを使用して、トポロジー内に存在するパスを判別し、予期しないパスの切断や予期しないパスの追加が発生した場所を判別します。パスの切断もパスの追加も、本番環境におけるトポロジーの対応する部分の根本原因分析に影響を及ぼします。

RCA パス・ツールは、指定されたエンティティー間の以下のタイプのパスを表示できます。

- 絶対パス: ネットワークの現行状態に関係なく、ソース・エンティティーとターゲット・エンティティーの間の最短パスが表示されます。絶対パス内のノードでイベントが発生した場合、そのパスは変わりません。例えば、パス上の 1 つ以上の中間デバイスで PingFail アラートなどのイベントが発生した場合、そのイベントは無視されます。RCA パス・ツールの照会では、絶対パスの設定は、`atoz.full` 表記によって示されます。

  注: 絶対パスを変更するには、トポロジーからエンティティーを削除する必要があります。これは重要な方針転換であり、詳細な調査を実施する場合に必要になることがあります。
- 現行パス: ライブ・パスとも呼ばれます。このパスには、ネットワークの現行状態を考慮した、ソース・エンティティーとターゲット・エンティティーの間の最短パスが表示されます。例えば、ターゲット・エンティティーへの最短パス上にある 1 つ以上の中間デバイスで PingFail アラートなどのイベントが発生した場合、ターゲット・エンティティーへのこのパスは中断され、RCA パス・ツールによって返されません。代替パスがある場合、代替パスの方がターゲット・エンティティーへの長いパスであっても、このパスが返されます。RCA パス・ツールの照会では、現在のパスの設定は `atoz.current` 表記によって示されます。

注: `atoz` データベースは実際には存在せず、`atoz.full` および `atoz.current` のどちらのテーブルも存在しません。

デバッグを実行するために最も効率的に RCA パス・ツールを使用する方法として、キャッシュからトポロジーをロードし、このキャッシュ・トポロジーに対して以下の調査アクティビティーを実行します。

1. トポロジー内の 2 つの対象ノード (A と Z) 間の既存のパス (現行パス) を判別します。
2. 対象のパス上にある A と Z の間の指定されたノードにイベントを注入します。
   a. A と Z の間に代替の現行パスは存在しますか?

b. パスはもうありませんか? 現行パスが存在しない場合、注入されたイベントは A または Z のイベントによって分離抑止されます。A と Z のどちらであるかは、ポーリング・プログラム・エンティティーの位置によって変わります。A であると想定すると、A のイベントは、A と Z の間のパスのイベントの根本原因を抑制している候補となります。

トポロジー・キャッシュのデバイスにイベントを注入するには、inject_fake_events.pl Perl スクリプトを使用します。inject_fake_events.pl Perl スクリプトについて詳しくは、「*IBM Tivoli Network Manager IP Edition* 管理ガイド」を参照してください。

注: RCA パス・ツールを、GUI で使用できるパス・ビュー・ツールと混同しないでください。RCA パス・ツールは、コマンド行ツールであり、主に根本原因分析のトラブルシューティングのために使用されます。一方で、GUI ベースのパス・ビューア・ツールは、選択された 2 つのデバイス間のネットワーク・パスを構成するデバイスとリンクのオペレーターに対してグラフィカル・ビューを表示します。

RCA パス・ツールの使用

以下の例では、RCA パス・ツールを使用して、ネットワーク上の指定されたソース・エンティティーとターゲット・エンティティーの間のパスを表示する方法について説明します。

使用例

RCA パス・ツールは、OQL サービス・プロバイダー (ncp_oql) を使用して照会を実行します。OQL サービス・プロバイダーについて詳しくは、「*IBM Tivoli Network Manager IP Edition* 管理ガイド」を参照してください。

以下のコマンドの例では、entityId フィールド値に 6 が指定されたソース・デバイスと entityId フィールド値に 137 が指定されたターゲット・デバイスとの間の絶対パスを照会します。

```
ncp_oql -domain NCOMS -service Events -query "select * from atoz.full
where a = 6 and z = 137;"
```

照会結果は、次のようになります。

```
{
   ENTITYID=6;
   ENTITYNAME='router4';
}

{
   ENTITYID=385;
   ENTITYNAME='VLAN_OBJECT_router4_VLAN_37';
}

{  ENTITYID=137;
   ENTITYNAME='router4[ Fa0/3/3 ]';
}

( 3 record(s) :   Transaction complete )
```

照会の例

特定のソース・エンティティーからのパスをトレースできます。あるいはオプションで、そのソース・エンティティー内に含まれている何らかのエンティティーから

のパスをトレースすることもできます。さらに、RCA パス・ツールの照会では、常に「a」というソース・エンティティーと「z」というターゲット・エンティティーの 2 つのエンティティーを指定する必要があります。これらの 2 つのエンティティーはパスのソースと宛先を表します。ソースとターゲットのエンティティーは、以下の任意の組み合わせとして指定できます。

- エンティティー ID
- エンティティー名
- IP アドレス

照会では、さらに、ソースに含まれている任意のエンティティーからパスをトレースできるように選択することができます。これは、収容シャーシからインターフェースへの直接パスが存在しない VLAN を扱う場合に役立ちます。

注: 照会は、イベント・ゲートウェイ・プロセスのトレース・ファイル ncp_g_event にデバッグ・レベル 1 で記録されます。例えば、照会ログ・ファイルの名前が ncp_g_event.NCOMS.trace である場合があります。

以下の照会は、RCA パス・ツールの使用方法の例を示しています。

1. entityId 102 から entityId 105 への絶対パスを表示します。

   ```
   ncp_oql -domain NCOMS -service Events -query "select * from atoz.full
   where a = 102 and z = 105;"
   ```

2. 「rod」という名前のエンティティーから「freddy」という名前のエンティティーへの絶対パスを表示します。

   ```
   ncp_oql -domain NCOMS -service Events -query "select * from atoz.full
   where a = 'rod' and z = 'freddy';"
   ```

3. entityId 102 から IP アドレス 172.21.226.3 を持つエンティティーへの現行パスを表示します。

   ```
   ncp_oql -domain NCOMS -service Events -query "select * from atoz.current
   where a = 102 and z = '172.21.226.3';"
   ```

4. 「rod[0 [1]]」という名前のインターフェースから entityId 105 への現行パスを表示します。

   ```
   ncp_oql -domain NCOMS -service Events -query "select * from atoz.current
   where a = 'rod[ 0 [ 1 ] ]' and z = 105;"
   ```

5. entityId 102 から IP アドレス 172.21.226.3 を持つエンティティーへの絶対パスを表示します。パスが検出されない場合は、entityId 102 のコンテナーに含まれている何らかのエンティティーからのパスの検出が試行されます。つまり、コンテナー階層の 1 つ上のレベルに移動してコンテナー ID を取得し、そのコンテナーに含まれている各エンティティーをソース・エンティティーとして使用してパスを構成しようとします。

   ```
   ncp_oql -domain NCOMS -service Events -query "select * from atoz.full
   where a = 102 and z = '172.21.226.3' and fromContained = 1;"
   ```

6. パスが存在しない場合は、そのことが出力で明確に示されます。

   ```
   ncp_oql -domain NCOMS -service Events -query "select * from atoz.full
   where a = 6 and z = 97;"
   ```

   パスが存在しない場合、出力は次のようになります。

```
{
    EntityId=0;
    EntityName='No path found from A to Z';
}
( 1 record(s) : Transaction complete )
```

例: パス上の潜在的な根本原因の判別

RCA パス・ツールを使用して、ネットワーク・パス上の障害をシミュレートすることができます。ターゲット・エンティティーへの代替パスがない場合、障害の下流にあるすべてのデバイスへのパスは中断されます。本番環境では、障害のあるデバイスに対応するデバイスが根本原因になります。

次の例では、デバイス A、B、C、および D が 1 列に接続されています。説明を簡単にするために、インターフェースは示されていません。

```
A ------------ B ------------- C ------------- D
```

Network Manager は、ポーリング・プログラム・エンティティーからデバイス D をポーリングします。次の図で、ポーリング・プログラム・エンティティーはエンティティー X として示されています。

```
X
\
 \
  \
   \
    A ------------ B ------------- C ------------- D
```

PingFail アラートがデバイス B に注入された場合、このアラートにより、ノード B が非アクティブになり、A から D へのパスが中断され、RCA パス・ツールは以下の結果を返します。

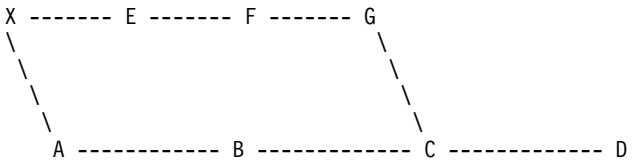
- 絶対パス **(atoz.full)**: ネットワークの現行状態に関係なく、ノード A と D の間の最短パスが表示されます。したがって、atoz.full には A から D へのパスが表示されます。
- 現行パス **(atoz.current)**: ネットワークの現行状態を考慮した、ノード A と D の間の最短パスが表示されます。ノード B が非アクティブであるため、A から D へのパスはありません。そのため、パスは返されません。

対応する本番環境では、PingFail アラートがノード D で発生した場合、このアラートは抑止され、ノード B のアラートは根本原因として強調表示されます。

例: 代替パスの判別

RCA パス・ツールを使用して、デバイス障害が発生した場合の代替ネットワーク・パスを判別することができます。ターゲット・エンティティーへの代替パスがある場合、本番環境では、障害のあるデバイスに対応するデバイスは根本原因になりません。ターゲット・エンティティーへの代替パスが存在するためです。

次の例では、1 列に接続されたノード D への 2 つのパスを含む、ネットワークの一部分について検討します。Network Manager は、ポーリング・プログラム・エンティティーからデバイス D をポーリングします。次の図で、ポーリング・プログラム・エンティティーはエンティティー X として示されています。説明を簡単にするために、インターフェースは示されていません。



PingFail アラートがデバイス B に注入された場合、このアラートにより、ノード B が非アクティブになり、A を経由する X から D へのパスが中断されます。ただし、E を経由する X から D への代替パスがあるため、RCA パス・ツールは以下の結果を返します。

- 絶対パス **(atoz.full)**: ネットワークの現行状態に関係なく、ノード X と D の間の最短パスが表示されます。その結果として、A を経由する X から D へのパスが表示されます。それが最短パスであるためです。
- 現行パス **(atoz.current)**: ネットワークの現行状態を考慮した、ノード X と D の間の最短パスが表示されます。その結果として、E を経由する X から D へのパスが表示されます。それが現行パスであるためです。ノード B は非アクティブであるため、A を経由する X から D へのパスは中断されています。

対応する本番環境では、PingFail アラートがノード D で発生した場合、このアラートは抑止されません。また、ノード B のアラートは根本原因として表示されません。

第 12 章 イベント・エンリッチの構成

イベントがイベント・ゲートウェイを通過するときにどのように処理されるかを構成することができます。

追加のイベント・エンリッチの構成

追加のイベント・エンリッチを実行するように、イベント・ゲートウェイを構成することができます。以下の例では、イベント・エンリッチを使用してイベントに追加できる情報の種類を示しています。

ObjectServer の alerts.status テーブルの任意のフィールドに取り込むように、イベント・ゲートウェイを構成することができます。既存のフィールドまたはカスタマイズしたフィールドに取り込むことができます。

注**:** イベント・ゲートウェイは alerts.status テーブルを変更しません。alerts.status テーブルに新しいフィールドを作成して、イベント・ゲートウェイにこの新しいフィールドに取り込ませるには、最初に ObjectServer で alerts.status テーブルを変更して新しいフィールドを追加する必要があります。

ObjectServer alerts.status テーブルに対する変更

イベント・ゲートウェイは alerts.status テーブルに新しいフィールドを作成しません。追加のイベント・エンリッチを構成する場合は、alerts.status テーブルに新しいフィールドを追加するように ObjectServer を構成する必要が生じる可能性があります。

以下は、カスタムのイベント・エンリッチを行う典型的な例を示しています。それぞれの例は、カスタムのイベント・エンリッチを構成する前に何らかの alerts.status テーブル構成が必要かどうかを示しています。

現在強化されていないデフォルトのイベント・フィールドの強化
: この一例として、PhysicalPort alerts.status フィールドを強化する場合が挙げられます。これは alerts.status テーブルにデフォルトで存在するフィールドであるため、ObjectServer を変更する必要はありません。

以前に別の目的で追加済みになっているカスタム・フィールドの強化
: この一例として、1 つ以上のプローブによって取り込まれたフィールドが既にあり、すべてのイベントに対してこのフィールドを取り込む場合が挙げられます。この例では、ポーリング・プログラムからモニター・プローブを介して到着した一部のイベントについて、NCIM キャッシュ・データに EXTRAINFO_sysLocation フィールドが取り込まれている可能性があります。 NmosLocation フィールドは既に ObjectServer に追加済みであり、可能な場合はモニター・プローブからこのフィールドが取り込まれます。これで、すべてのイベントに対してこのフィールドを取り込むことができます。この場合は、ObjectServer を変更する必要はありません。

NCIM トポロジー・データベースからのトポロジー・エンリッチの実行
: この場合は、NCIM からの任意のデータを使用してイベントを強化しよう

© Copyright IBM Corp. 2006, 2016

としています。最初に ObjectServer を変更して、alerts.status テーブルに新しいフィールドを 1 つ以上追加する必要があります。

例: メイン・ノード・デバイスのロケーションによるイベントの強化

イベントに関連付けられているメイン・ノード・デバイスのロケーションがイベントのフィールドに追加されるように、イベント・エンリッチを構成することができます。

ObjectServer のどのフィールドに取り込むかを検討します。alerts.status テーブルには既にデフォルトの Location フィールドがあります。このフィールドへの取り込みがまだ行われていない限り、この例ではこのフィールドに取り込むことを前提とします。何らかの理由により、強化されたロケーション値を格納するためのカスタマイズ・フィールドを別個に作成する場合は、メイン・ノード・デバイスのロケーションを格納するためのフィールド (例えば、NmosLocation) を alerts.status テーブルに追加することができます。 ObjectServer テーブルにカスタム・フィールドを追加する方法については、*IBM Tivoli Netcool/OMNIbus* 管理ガイドを参照してください。

イベントに関連付けられているメイン・ノード・デバイスのロケーションは、NCIM トポロジー・データベースのシャーシ・テーブルにあります。このフィールドは、NCIM キャッシュを使用してアクセス可能であり、ncimCache.entityData テーブルに保持されます。

NCIM キャッシュのテーブルおよびフィールドの構造について詳しくは、*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンスを参照してください。

以下のステップで、この追加のイベント・エンリッチを構成する方法を説明します。

1. イベント・ゲートウェイ・スキーマ・ファイル `$NCHOME/etc/precision/EventGatewaySchema.cfg` を編集して、イベント・ゲートウェイが新しいフィールドを更新できるようにします。これを行うには、以下の太字のテキストを出力イベント・フィルターに追加します。新しい `Location` フィールドを含む行の前の `NmosSerial` フィールドを含む行の末尾には、必ずコンマを追加してください。

```
insert into config.ncp2nco
(
    FieldFilter
)
values
(
    [
        "NmosCauseType",
        "NmosDomainName",
        "NmosEntityId",
        "NmosManagedStatus",
        "NmosObjInst",
        "NmosSerial",
        "Location"
    ]
);
```

注: 出力イベント・フィルターに追加されるフィールドは自動的に着信フィールド・フィルター config.nco2ncp に追加されるため、このフィールドの現行値が確実に取得されます。これにより次のステップで、StandardEventEnichment スティッチャーが InterfaceName フィールドを更新する前にこのフィールドの値を確認できます。この技法を使用すると、イベント・ゲートウェイが同じ値を更新し続けることがなくなります。

2. イベント・ゲートウェイ・スティッチャーを編集して、トポロジー・データベースからロケーション情報を取得し、Location フィールドに取り込みます。 そのための方法の 1 つは、次のコードを StandardEventEnichment スティッチャーに追加することです。このコードを追加すると、エンティティーと一致するすべてのトポロジー・イベントに対して必ずこの手順が実行されます。このスティッチャーへのコード追加は、最終行の `GwEnrichEvent( enrichedFields )` の呼び出し直前に行います。`GwEnrichEvent()` スティッチャー・ルールについて詳しくは、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。

表 *46.* メイン・ノード・デバイスのロケーションに関連するコードの行の例

行番号	説明
1	`GwMainNodeLookupUsing()` ルールを呼び出して、現在のイベントのシャーシ・データが使用可能であるか確認します。イベントはインターフェースで生成された可能性があり、その場合、通常はこの時点でシャーシ・データは使用可能ではありません。 `GwMainNodeLookupUsing()` スティッチャー・ルールについて詳しくは、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。
5	シャーシ・テーブルから sysLocation データを取得します。 **注:** NCIM キャッシュからデータを取得するときは、エンティティー・データのフィールドを大文字で指定してください (例えば `@mainNode.chassis.SYSLOCATION` とします)。
7 - 9	Location フィールドがまだ設定されていない場合は、強化する他のフィールドに sysLocation データを追加します。

```
1]    Record mainNode = GwMainNodeLookupUsing( "LocalNodeAlias" );
2]
3]    if ( mainNode <> NULL )
4]    {
5]          text sysLocation = @mainNode.chassis.SYSLOCATION;
6]
7]          if ( sysLocation <> eval(text, '&Location') )
8]          {
9]               @enrichedFields.Location = sysLocation;      }
10]         }
```

関連概念:

107 ページの『発信フィールド・フィルター』
発信フィールド・フィルターは、イベント・ゲートウェイによって更新される可能性のある ObjectServer フィールド・セットを定義します。

関連資料:

134 ページの『例: StandardEventEnrichment.stch』
このトピックでは、イベント・エンリッチ・スティッチャーの仕組みについて説明します。

例: インターフェース名によるイベントの強化

すべてのインターフェース・イベントに関してイベントが発生したインターフェースの名前がそのイベントのフィールドに追加されるように、イベント・エンリッチを構成することができます。

強化されたインターフェース名の値を格納するために、ObjectServer の alerts.status テーブルに新しいカスタム・フィールドを作成する必要があります。この例では、alerts.status テーブルに InterfaceName という新しいカスタム・フィールドが作成済みであることを前提としています。ObjectServer テーブルにカスタム・フィールドを追加する方法については、*IBM Tivoli Netcool/OMNIbus* 管理ガイドを参照してください。

インターフェースの名前は、NCIM トポロジー・データベース・インターフェース・テーブル内にあります。このフィールドは、NCIM キャッシュを使用してアクセス可能であり、ncimCache.entityData テーブルに保持されます。

NCIM キャッシュのテーブルおよびフィールドの構造について詳しくは、*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンスを参照してください。

以下のステップで、この追加のイベント・エンリッチを構成する方法を説明します。

1. イベント・ゲートウェイ・スキーマ・ファイル `$NCHOME/etc/precision/EventGatewaySchema.cfg` を編集して、イベント・ゲートウェイが新しいフィールドを更新できるようにします。これを行うには、以下の太字のテキストを出力イベント・フィルターに追加します。新しい `InterfaceName` フィールドを含む行の前の `NmosSerial` フィールドを含む行の末尾には、必ずコンマを追加してください。

   ```
   insert into config.ncp2nco
   (
       FieldFilter
   )
   values
   (
       [
           "NmosCauseType",
           "NmosDomainName",
           "NmosEntityId",
           "NmosManagedStatus",
           "NmosObjInst",
           "NmosSerial",
           "InterfaceName"
       ]
   );
   ```

   注: 出力イベント・フィルターに追加されるフィールドは自動的に着信フィールド・フィルター config.nco2ncp に追加されるため、このフィールドの現行値が確実に取得されます。これにより次のステップで、StandardEventEnichment スティッチャーが InterfaceName フィールドを更新する前にこのフィールドの値を確認できます。この技法を使用すると、イベント・ゲートウェイが同じ値を更新し続けることがなくなります。

2. イベント・ゲートウェイ・スティッチャーを編集して、トポロジー・データベースからインターフェース名情報を取得し、InterfaceName フィールドに取り込

みます。 そのための方法の 1 つは、次のコードを StandardEventEnichment スティッチャーに追加することです。このコードを追加すると、エンティティーと一致するすべてのトポロジー・イベントに対して必ずこの手順が実行されます。このスティッチャーへのコード追加は、entityType 値の確認の後の、最終行の GwEnrichEvent(enrichedFields) の呼び出し直前に行います。 GwEnrichEvent() スティッチャー・ルールについて詳しくは、*IBM Tivoli Network Manager IP Edition* 言語リファレンス を参照してください。

表 *47.* インターフェース名に関連するコードの行の例

行番号	説明
1	このイベント・エンリッチは、インターフェース・イベントのみに関連しています。 entityType 値が 2 であることを確かめることによってこのイベントがインターフェースに関連することを確認し、この確認ができたら処理を続行します。
3	インターフェース・テーブルから ifName データを取得します。 **注:** NCIM キャッシュからデータを取得するときは、エンティティー・データのフィールドを大文字で指定してください (例えば @mainNode.chassis.SYSLOCATION とします)。
5 - 8	インターフェース名の値がまだスコープ内イベントに存在していない場合にのみ、InterfaceName フィールドに取り込みます。

```
1]    if ( entityType == 2 )
2]    {
3]         text interfaceName = @entity.interface.IFNAME;
4]
5]        if ( interfaceName <> eval(text, '&InterfaceName') )
6]        {
7]              @enrichedFields.InterfaceName = interfaceName;
8]        }
9]    }
```

関連概念:

107 ページの『発信フィールド・フィルター』
発信フィールド・フィルターは、イベント・ゲートウェイによって更新される可能性のある ObjectServer フィールド・セットを定義します。

関連資料:

134 ページの『例: StandardEventEnrichment.stch』
このトピックでは、イベント・エンリッチ・スティッチャーの仕組みについて説明します。

ObjectServer 更新間隔フィールドの構成

イベント・ゲートウェイが ObjectServer に対するイベント・エンリッチの更新をキューに入れる際の間隔を構成することができます。

ObjectServer の更新間隔のデフォルト設定は 5 秒です。使用しているシステムのデータ・フローに合わせて、この値を変更することをお勧めします。

- 1 回の ObjectServer 更新としてグループ化するイベント・エンリッチの更新の数を増やすには、この値を大きくします。これにより、ObjectServer への負荷は軽減されますが、ObjectServer でのイベント・エンリッチの更新における遅延は大きくなります。
- ObjectServer に対するイベント・エンリッチの更新を高速化するには、この値を小さくします。この場合、ObjectServer はより多くのイベント・エンリッチの更新を管理しなければならなくなるため、その負荷は増えます。

イベント・ゲートウェイ用の構成ファイルは EventGatewaySchema.cfg 構成ファイルです。このファイルの場所は、$NCHOME/etc/precision/EventGatewaySchema.cfg です。ObjectServer の更新間隔は、config.defaults テーブルの ObjectServerUpdateInterval フィールドに格納されています。

1. EventGatewaySchema.cfg 構成ファイルを開きます。
2. config.defaults テーブルに対する insert ステートメントを確認します。 デフォルトでは、この insert ステートメントの形式は以下のとおりです。

```
insert into config.defaults
(
    IDUCFlushTime,
    ObjectServerUpdateInterval,
    NcpServerEntity
)
values
(
    5,
    5,
    ""
);
```

   デフォルトでは ObjectServerUpdateInterval フィールドは 5 秒に設定されています。
3. ObjectServerUpdateInterval フィールドの値を目的の値 (秒単位) に変更します。

関連概念:

109 ページの『出力イベント・ゲートウェイ・キュー』
出力イベント・ゲートウェイ・キューは、イベント・ゲートウェイ・スティッチャー (主となるイベント・エンリッチ) およびプラグインから、強化されたイベントを受け取ります。 更新の数を最小限にして ObjectServer への負荷を最小限に抑えるために、Object Server への更新はキューに入れられ、集約されたうえで、指定した間隔で ObjectServer に送信されます。 デフォルトは 5 秒です。

関連資料:

270 ページの『config.defaults テーブル』
config.defaults テーブルには、イベント・ゲートウェイに関する一般構成データが入っています。

OQL サービス・プロバイダーを使用してイベント・ゲートウェイ・データベースにログインする

オブジェクト照会言語 (OQL) サービス・プロバイダーおよび EventGateway サービス名を使用してデータベースにログインし、ゲートウェイ・データベースを照会する必要があります。

以下のコマンド行の例では、NCOMS ドメインで稼働している、イベント・ゲートウェイの NcoGate サービスにログインしています。

```
ncp_oql -domain NCOMS -service EventGateway
```

OQL サービス・プロバイダーのユーザー認証は、デフォルトではオフになっています。認証がオンになっている場合は、プロンプトで有効なユーザー名とパスワードを入力します。

ObjectServer の照会

OQL サービス・プロバイダーを使用して、ObjectServer を照会することができます。

以下に示す OQL サービス・プロバイダーのコマンド行の例では、NCOMS と呼ばれる ObjectServer 上の NCOMS ドメインで稼働している ObjectServer サービスにログインしています。

```
ncp_oql -domain NCOMS -service ObjectServer -server NCOMS -username netcool
```

OQL サービス・プロバイダーのユーザー認証は、デフォルトではオフになっています。認証がオンになっている場合は、プロンプトで有効なユーザー名とパスワードを入力します。

注: `-server` 引数はオプションです。この引数を指定しない場合は、`$NCHOME/etc/precision/ConfigItnm.cfg` ファイルで構成されているサーバーが使用されます。

NCIM データベースの照会

OQL サービス・プロバイダーを使用して、NCIM データベースを照会することができます。

以下に示す OQL サービス・プロバイダーのコマンド行の例では、NCOMS ドメインで稼働している、NCIM サービス内の NCMONITOR スキーマにログインしています。これは、NCMONITOR スキーマ内のテーブル (activeEvent テーブルなど) にアクセスする場合に便利です。

```
ncp_oql -domain NCOMS -service Ncim -dbId NCMONITOR
```

OQL サービス・プロバイダーのユーザー認証は、デフォルトではオフになっています。認証がオンになっている場合は、プロンプトで有効なユーザー名とパスワードを入力します。

注: `-dbId` 引数はオプションです。

イベントと ObjectServer の再同期化

イベント・ゲートウェイに対して SIGHUP コマンドを実行すると、イベント・ゲートウェイの構成を変更できます。

コマンド `kill -HUP PID` を入力します (*PID* はイベント・ゲートウェイのプロセス ID です)。

イベント・ゲートウェイは、構成ファイルのタイム・スタンプをチェックします。構成ファイルが変更されていれば、イベント・ゲートウェイはその構成ファイルを再び読み取って、構成変更を処理します。

注**:** このコマンドは、すべてのイベントをイベント・ゲートウェイと再同期させる処理も行います。

SIGHUP コマンドの処理ステップ

SIGHUP コマンドの処理を、以下のステップで説明します。

1. イベント・ゲートウェイが HUP コマンドを受信します。
2. イベント・ゲートウェイが ObjectServer IDUC チャネルでのイベントの listen を停止します。
3. イベント・ゲートウェイが、自身の現在のイベント・キャッシュを空にします。このキャッシュは、イベント状態を判別するために使用されます。
4. イベント・ゲートウェイが、そのすべてのプラグインに合成再同期開始イベントを送信します。
5. RCA プラグインが mojo.events データベース表を空にし、NCIM キャッシュ内のデータに基づいてグラフを再描画します。

   注**:** RCA プラグインは、この時点で RCASchema.cfg 構成ファイルまたは RCA スティッチャーを再読み取りしません。
6. イベント・ゲートウェイが、始動時と同じ方法で Object Server からすべてのイベントを取得します。
7. イベント・ゲートウェイが、始動時と同じ方法ですべてのイベントを処理し、関連するイベントをプラグインに渡します。
8. イベント・ゲートウェイが、再同期終了イベントをそのプラグインに送信します。
9. イベント・ゲートウェイが ObjectServer IDUC チャネルでのイベントの listen を再開します。

関連タスク:

201 ページの『第 14 章 根本原因分析の構成』
RCA プラグインを構成できます。

共通イベント・ゲートウェイ・プロパティーの構成

NCP_G_EVENT.props ファイルを編集して、共通イベント・ゲートウェイ・プロパティーを構成できます。

Tivoli Netcool/OMNIbus ゲートウェイには、多くの共通プロパティーおよびそれらに関連するコマンド行オプションがあります。プロパティーで定義されるのは、汎用機能の設定 (メッセージ・ロギングなど)、プロセス間通信 (IPC) の設定、および共通ゲートウェイの設定 (クライアントがサーバーの応答を待機するタイムアウト期間の設定など) です。

Network Manager イベント・ゲートウェイは、Tivoli Netcool/OMNIbus ゲートウェイ共通プロパティーのデフォルト値を使用します。$NCHOME/etc/precision ディレクトリー内にインストールされている NCP_G_EVENT.props プロパティー・ファイルを編集して、Tivoli Netcool/OMNIbus ゲートウェイ共通プロパティーを他の値に構成できます。例えば、タイムアウトを回避するために、Ipc.Timeout 共通プロパティーにデフォルト (60 秒) 以外の値を指定できます。これにより、アクティビティーが検出されなかった場合にプライマリー・ ObjectServer への接続を素早く除去できます。

現時点では、構成可能な Tivoli Netcool/OMNIbus ゲートウェイ共通プロパティーは Ipc.Timeout のみです。Ipc.Timeout プロパティーは、クライアントがサーバーの応答を待機する期間 (秒) を指定します。この時間を超えると、エラーがログ記録されます。デフォルトは 60 秒です。

Ipc.Timeout プロパティーを構成するには、以下のようにします。

1. $NCHOME/etc/precision ディレクトリー内にインストールされている NCP_G_EVENT.props プロパティー・ファイルをバックアップします。
2. テキスト・エディターで、NCP_G_EVENT.props プロパティー・ファイルを開きます。
3. Ipc.Timeout プロパティーを見つけ、デフォルト値の 60 秒をご使用のネットワークに適した値に置き換えます。
4. NCP_G_EVENT.props プロパティー・ファイルを保存します。

以下の例では、Ipc.Timeout プロパティーを値 20 (秒) に構成します。

```
# INTEGER (IPC Session timeout), default 60 seconds
Ipc.Timeout: 20
```

第 13 章 イベント・ゲートウェイ・プラグインの構成

イベント・ゲートウェイ・プラグインを構成することができます。さらに、現在有効になっているプラグインを表示することもできます。

プラグインの有効化と無効化

プラグインを有効または無効にするには、ncp_gwplugins.pl スクリプトを使用します。プラグインごとにスクリプトを個別に実行します。

$NCHOME/precision/scripts/perl/scripts/ncp_gwplugins.pl スクリプトを実行します。 コマンド行オプションを使用して、ドメインとプラグインを指定し、プラグインを無効にするか有効にするかを指定します。コマンド行オプションについては、以下の表で説明しています。

表 *48. ncp_gwplugins.pl* コマンド行オプション

コマンド行オプション	説明
-domain *DomainName*	必須: プラグインに関連するドメインの名前。このドメインを使用することにより、このスクリプトは関連のイベント・ゲートウェイ・プラグイン・データベースに接続して更新するために、関連の DbLogins.cfg ファイルを読み取ることができます。
-plugin *PluginName*	プラグインの名前。 **注:** スクリプトを実行できるのは、一度に 1 つのプラグインについてのみです。このコマンド行オプションで使用するプラグイン名は、以下のとおりです。プラグイン名が複数の単語から構成される場合は、名前を二重引用符で囲んでください (例えば "Adaptive Polling" とします)。 • Adaptive Polling • Disco • Failover • Fix Pack 4 PostNcimProcessing • RCA • SAE IP Path • SAE ITNM Service • SAE MPLS VPN • zNetView
-disable	指定したプラグインを無効にします。
-enable	指定したプラグインを有効にします。

© Copyright IBM Corp. 2006, 2016

表 48. *ncp_gwplugins.pl* コマンド行オプション (続き)

コマンド行オプション	説明
-global	すべてのドメインでプラグインを有効にします。 これを指定しない場合、-domain オプションによって指定したドメイン内のみでプラグインが有効になります。
-help	コマンド行オプションのフル・セットのヘルプ・テキストを表示します。 **ヒント:** 使用可能なオプションの簡単なリストを表示するには、オプションを指定せずにスクリプトを実行します。

例えば、すべてのドメインで zNetView プラグインを有効にするには、次のようにスクリプトを実行してください。

```
$NCHOME/precision/bin/ncp_perl $NCHOME/precision/scripts/perl/scripts/
ncp_gwplugins.pl -domain NCOMS -plugin zNetView -enable -global
```

関連資料:

140 ページの『プラグインの説明』
ここでは、各イベント・ゲートウェイ・プラグインの仕組みについて説明します。

280 ページの『ncmonitor 内の ncp_g_event プラグイン・データベース表』
ここでは、ncmonitor データベース内にあるイベント・ゲートウェイ構成表と、各表に含まれている情報の種類について説明します。 これらの表のほとんどは、イベント・ゲートウェイのプラグイン構成に関係します。

プラグイン情報のリスト

イベント・ゲートウェイ・プラグインに関する情報をリストすることができます。例えば、各プラグインがサブスクライブしているイベント・マップとイベント状態をリストすることができます。

プラグイン情報をリストするには ncp_gwplugins.pl スクリプトを使用します。このスクリプトは $NCHOME/precision/scripts/perl/scripts/ncp_gwplugins.pl にあります。

このスクリプトを実行してイベント・マップのサブスクリプションをリストするには、次のようなコマンドを実行します。この例では、Disco プラグインがサブスクライブしているイベント・マップとイベント状態をすべてリストします。

```
$NCHOME/precision/bin/ncp_perl
$NCHOME/precision/scripts/perl/scripts/ncp_gwplugins.pl -domain NCOMS -plugin Disco
```

コマンド行オプション

この例で使用されている ncp_gwplugins.pl スクリプトのコマンド行オプションについて、次の表で説明します。 ヘルプを参照するには、次のようにスクリプトを実行してください。

- 使用可能なオプションの簡単なリストを表示するには、オプションを指定せずにコマンドを実行します。
- フル・セットのコマンド行オプションを表示するには、-help オプションを指定してスクリプトを実行します。

表 49. *ncp_gwplugins.pl* コマンド行オプション

コマンド行オプション	説明
`-domain` *DomainName*	必須。プラグインに関連するドメインの名前。 このドメインを使用することにより、このスクリプトは関連のイベント・ゲートウェイ・プラグイン・データベースに接続するために、関連の DbLogins.cfg ファイルを読み取ることができます。
`-plugin` *PluginName*	プラグインの名前。 **注:** スクリプトを実行できるのは、一度に 1 つのプラグインについてのみです。このコマンド行オプションで使用するプラグイン名は、以下のとおりです。プラグイン名が複数の単語から構成される場合は、名前を二重引用符で囲んでください (例えば `"Adaptive Polling"` とします)。 • Adaptive Polling • Disco • Failover • Fix Pack 4 PostNcimProcessing • RCA • SAE IP Path • SAE ITNM Service • SAE MPLS VPN • zNetView

関連資料:

140 ページの『プラグインの説明』
ここでは、各イベント・ゲートウェイ・プラグインの仕組みについて説明します。

280 ページの『ncmonitor 内の ncp_g_event プラグイン・データベース表』
ここでは、ncmonitor データベース内にあるイベント・ゲートウェイ構成表と、各表に含まれている情報の種類について説明します。 これらの表のほとんどは、イベント・ゲートウェイのプラグイン構成に関係します。

イベント・マップ・サブスクリプションの変更

プラグインがサブスクライブしているイベント・マップを変更することができます。例えば、新しいイベント・マップを追加し、そのイベント・マップによって処理されるイベントに対してシステムによる RCA を実行する場合は、そのイベント・マップを RCA プラグインのサブスクリプション・リストに追加する必要があります。

イベント・マップ・サブスクリプションを変更するには、`ncp_gwplugins.pl` スクリプトを使用します。 このスクリプトは `$NCHOME/precision/scripts/perl/scripts/ncp_gwplugins.pl` にあります。

このスクリプトを実行してイベント・マップ・サブスクリプションを変更するには、次のようなコマンドを実行します。 この例では、PnniIfState というイベント・マップを RCA プラグインのサブスクリプション・リストに追加します。

```
$NCHOME/precision/perl/bin/ncp_perl $NCHOME/precision/scripts/
perl/scripts/ncp_gwplugins.pl -domain NCOMS -plugin RCA -add -eventMap PnniIfState
```

コマンド行オプション

この例で使用されている ncp_gwplugins.pl スクリプトのコマンド行オプションについて、次の表で説明します。 ヘルプを参照するには、次のようにスクリプトを実行してください。

- 使用可能なオプションの簡単なリストを表示するには、オプションを指定せずにコマンドを実行します。
- フル・セットのコマンド行オプションを表示するには、-help オプションを指定してスクリプトを実行します。

表 *50. ncp_gwplugins.pl* コマンド行オプション

コマンド行オプション	説明
-domain *DomainName*	必須。プラグインに関連するドメインの名前。 このドメインを使用することにより、このスクリプトは関連のイベント・ゲートウェイ・プラグイン・データベースに接続して更新するために、関連の DbLogins.cfg ファイルを読み取ることができます。
-plugin *PluginName*	プラグインの名前。 **注:** スクリプトを実行できるのは、一度に 1 つのプラグインについてのみです。このコマンド行オプションで使用するプラグイン名は、以下のとおりです。プラグイン名が複数の単語から構成される場合は、名前を二重引用符で囲んでください (例えば "Adaptive Polling" とします)。 • Adaptive Polling • Disco • Failover • Fix Pack 4 PostNcimProcessing • RCA • SAE IP Path • SAE ITNM Service • SAE MPLS VPN • zNetView
-add	指定した 1 つ以上のプラグインについて、指定したイベント・マップへの関心を追加します。 -plugin オプションと -eventMap オプションを指定する必要があります。

表 *50. ncp_gwplugins.pl* コマンド行オプション *(*続き*)*

コマンド行オプション	説明
`-drop`	指定した 1 つ以上のプラグインについて、指定したイベント・マップへの関心を削除します。
`-eventMap` *EventMapName*	関心を追加または削除するイベント・マップ。

関連概念:

99 ページの『イベント・エンリッチのクイック・リファレンス』
ここでは、イベントがイベント・ゲートウェイを通過するときにどのように処理されるかを説明します。

155 ページの『RCA のクイック・リファレンス』
ここでは、イベントが RCA プラグインを通過するときにどのように処理されるかを説明します。

109 ページの『出力イベント・ゲートウェイ・キュー』
出力イベント・ゲートウェイ・キューは、イベント・ゲートウェイ・スティッチャー (主となるイベント・エンリッチ) およびプラグインから、強化されたイベントを受け取ります。 更新の数を最小限にして ObjectServer への負荷を最小限に抑えるために、Object Server への更新はキューに入れられ、集約されたうえで、指定した間隔で ObjectServer に送信されます。 デフォルトは 5 秒です。

関連資料:

140 ページの『プラグインの説明』
ここでは、各イベント・ゲートウェイ・プラグインの仕組みについて説明します。

280 ページの『ncmonitor 内の ncp_g_event プラグイン・データベース表』
ここでは、ncmonitor データベース内にあるイベント・ゲートウェイ構成表と、各表に含まれている情報の種類について説明します。 これらの表のほとんどは、イベント・ゲートウェイのプラグイン構成に関係します。

プラグイン構成パラメーターの設定

`ncp_gwplugins.pl` スクリプトを使用して、イベント・ゲートウェイ・プラグインのオプションの構成パラメーターを設定することができます。

オプションの構成パラメーターを設定するには、`ncp_gwplugins.pl` スクリプトを使用します。このスクリプトは `$NCHOME/precision/scripts/perl/scripts/ncp_gwplugins.pl` にあります。

このスクリプトを実行して構成パラメーターを設定するには、次のようなコマンドを実行します。この例では、ncmonitor.activeEvent テーブルの更新間隔を 10 秒に設定します。デフォルトは 5 秒です。

```
$NCHOME/precision/perl/bin/ncp_perl
$NCHOME/precision/scripts/perl/scripts/ncp_gwplugins.pl -domain NCOMS
-plugin "Adaptive Polling" -set -name ActiveEventUpdateInterval -value 10
```

コマンド行オプション

この例で使用されている ncp_gwplugins.pl スクリプトのコマンド行オプションについて、次の表で説明します。 ヘルプを参照するには、次のようにスクリプトを実行してください。

- 使用可能なオプションの簡単なリストを表示するには、オプションを指定せずにコマンドを実行します。
- フル・セットのコマンド行オプションを表示するには、-help オプションを指定してスクリプトを実行します。

表 51. *ncp_gwplugins.pl* コマンド行オプション

コマンド行オプション	説明
-domain *DomainName*	必須: プラグインに関連するドメインの名前。ドメインは、スクリプトがどの DbLogins.*domain*.cfg を読み取ればよいかを示します。これにより、スクリプトは、正しいイベント・ゲートウェイ・プラグイン・データベースに接続し、データベースを更新できます。
-plugin *PluginName*	プラグインの名前。 **注:** スクリプトを実行できるのは、一度に 1 つのプラグインについてのみです。このコマンド行オプションで使用するプラグイン名は、以下のとおりです。プラグイン名が複数の単語から構成される場合は、名前を二重引用符で囲んでください (例えば "Adaptive Polling" とします)。 • Adaptive Polling • Disco • Failover • Fix Pack 4 PostNcimProcessing • RCA • SAE IP Path • SAE ITNM Service • SAE MPLS VPN • zNetView
-set	変数を設定することを示します。
-name *ParameterName*	設定するパラメーターの名前。
-value *Parametervalue*	このパラメーターに設定する値。

関連資料:

140 ページの『プラグインの説明』
ここでは、各イベント・ゲートウェイ・プラグインの仕組みについて説明します。

280 ページの『ncmonitor 内の ncp_g_event プラグイン・データベース表』
ここでは、ncmonitor データベース内にあるイベント・ゲートウェイ構成表と、各表に含まれている情報の種類について説明します。 これらの表のほとんどは、イベ

ント・ゲートウェイのプラグイン構成に関係します。

SAE プラグインの構成

ここでは、SAE プラグインの構成方法を説明します。

関連概念:

148 ページの『SAE プラグイン』

SAE プラグインは、MPLS VPN および IP パスのサービスに影響を与えるイベントを生成します。

サービスに影響を与えるイベントの要約フィールド情報の構成

サービスに影響を与えるイベントがオペレーターにとってより有益になるように、SAE プラグインを構成して、サービスに影響を与えるイベントの「要約」フィールドにカスタマー関連情報を挿入できます。

この変更を行う、SAE プラグイン内の構成ファイルを以下に示します。

- IP パス・サービス用の SaeIpPath.cfg (`$NCHOME/etc/precision/SaeIpPath.cfg`)
- MPLS VPN サービス用の SaeMplsVpn.cfg (`$NCHOME/etc/precision/SaeMplsVpn.cfg`)
- カスタム・サービス用の SaeItnmService.cfg (`$NCHOME/etc/precision/SaeItnmService.cfg`)

追加情報を構成して SAE の「要約」フィールドに挿入するために、これらの各ファイルで使用されるフィールドは、CustomerNameField と呼ばれます。以下の例は、SaeMplsVpn.cfg ファイルでこのフィールドを構成する方法を示しています。

1. SaeMplsVpn.cfg 構成ファイルを開きます。
2. 太字で示されたテキストを追加して insert ステートメントを変更し、NCIM キャッシュ内のサービス・レコードの関連フィールドから、CustomerNameField フィールドにデータを挿入します。 例えば、以下のステートメントでは、entityData->DESCRIPTION フィールド (このフィールドが存在する場合) の内容を CustomerNameField、および生成された MPLS VPN エッジ・サービスの SAE の「要約」フィールドに挿入します。

   注**:** insert ステートメントにフィールドを追加する場合、前の行にコンマを追加する必要があります。

```
insert into config.serviceTypes
(
    ServiceTypeName,
    CollectionEntityType,
    ConstraintFilter,
    CustomerNameField
)
values
(
    "MPLSVPNEdgeService",
    17 -- "networkVpn",
    "networkVpn->VPNTYPE <> 'MPLS Core'",
    "entityData->DESCRIPTION"
```

SAE プラグインへの SAE タイプの追加

デフォルトで提供されている 3 つのタイプ以外の SAE を生成するように、SAE プラグインを構成することができます。例えば、MPLS VPN エッジ・エンティティーの SAE イベント (あるタイプの SAE) と、MPLS VPN コア・エンティティーの SAE イベント (別のタイプの SAE) を作成するように、このプラグインを構成することができます。

この例では、既存の構成ファイル SaeMplsVpn.cfg をカスタマイズして、config.serviceTypes テーブルに MPLS VPN SAE サービス・タイプを追加します。新しいサービス・タイプは MPLS VPN Core Service と呼ばれ、コア・ネットワーク内のいずれかのルーターで重大度 5 (重大) の障害イベントが発生すると SAE を生成します。まったく新しい構成ファイルを作成し、そこに関連の insert を指定することにより、新しい SAE サービス・タイプを作成することもできます。

SAE プラグインの MPLS VPN SAE サービス・タイプ用の構成ファイルは SAEMplsVpn.cfg 構成ファイルです。このファイルの場所は、$NCHOME/etc/precision/SAEMplsVpn.cfgです。

1. SAEMplsVpn.cfg 構成ファイルを開きます。
2. デフォルトの insert が MPLS VPN Edge Service を作成します。次のように記述されます。

```
insert into config.serviceTypes
(
    ServiceTypeName,
    CollectionEntityType,
    ConstraintFilter
)
values
(
    "MPLS VPN Edge Service",
    17,   -- networkVpn
    "networkVpn->VPNTYPE <> 'MPLS Core'"
);
```

3. 既存の insert の後に新しい insert を追加します。新しい insert は次のように記述されるはずです。

```
insert into config.serviceTypes
(
    ServiceTypeName,
    CollectionEntityType,
    ConstraintFilter
)
values
(
    "MPLS VPN Core Service",
    17,   -- networkVpn
    "networkVpn->VPNTYPE = 'MPLS Core'"
);
```

注: この例に示すように、1 つの特定のテーブル (例えば、networkVpn (17)) に 2 つ以上の SAE サービス・タイプを含めることができます。この場合、SAE サービス・タイプは相互に排他的な組み合わせでなければなりません。そうでないと、これらがオーバーラップした場合に一方が他方を取り込むことにな

ります。例えば、この例で説明するサービス・タイプは、次のように補完的な ConstraintFilter 設定になっているため、オーバーラップすることはありません。

- networkVpn->VPNTYPE <> 'MPLS Core'
- networkVpn->VPNTYPE = 'MPLS Core'

関連概念:

148 ページの『SAE プラグイン』

SAE プラグインは、MPLS VPN および IP パスのサービスに影響を与えるイベントを生成します。

第 14 章 根本原因分析の構成

RCA プラグインを構成できます。

関連資料:

188 ページの『イベントと ObjectServer の再同期化』
イベント・ゲートウェイに対して SIGHUP コマンドを実行すると、イベント・ゲートウェイの構成を変更できます。

ポーリング・プログラム・エンティティーの構成

Network Manager サーバーがご使用のネットワーク・ドメインのスコープ内にない場合に RCA プラグインが分離抑止を実行できるようにするには、入口インターフェースの IP アドレスまたは DNS 名をポーリング・プログラム・エンティティーとして指定します。

イベント・ゲートウェイ用の構成ファイルは EventGatewaySchema.cfg 構成ファイルです。このファイルの場所は、`$NCHOME/etc/precision/EventGatewaySchema.cfg` です。ポーリング・プログラム・エンティティーの値は、config.defaults テーブルの NcpServerEntity フィールドに格納されています。

1. EventGatewaySchema.cfg 構成ファイルを開きます。
2. config.defaults テーブルに対する insert ステートメントを確認します。 デフォルトでは、この insert ステートメントの形式は以下のとおりです。

```
insert into config.defaults
(
    IDUCFlushTime,
    ObjectServerUpdateInterval,
    NcpServerEntity
)
values
(
    5,
    5,
    ""
);
```

   デフォルトでは NcpServerEntity フィールドは空です。この場合、イベント・ゲートウェイは、それが稼働されているローカル・ホストの 1 つ以上の IP アドレスを使用してトポロジーを検索します。
3. このステートメントを変更して、NcpServerEntity フィールドを入口インターフェースの IP アドレスまたは DNS 名の値に設定します。

関連概念:

158 ページの『ポーリング・プログラム・エンティティー』
ここでは、ポーリング・プログラム・エンティティーとは何であるか、およびその構成方法を説明します。

関連資料:

© Copyright IBM Corp. 2006, 2016

Fix Pack 4 203 ページの『クロスドメイン・ネットワークでの RCA の考慮事項』
クロスドメイン環境では、各ディスカバリー・ドメインの **ncp_g_event** プロセスは、同じディスカバリー・ドメイン内のデバイスに対して RCA を実行します。各ドメイン内では、RCA は、単一のドメインしかない場合と同様に動作します。複数のドメインがクロスドメイン・ディスカバリーを使用して一緒に視覚化されている場合、複数のドメインで根本原因を分析することもできます。

270 ページの『config.defaults テーブル』
config.defaults テーブルには、イベント・ゲートウェイに関する一般構成データが入っています。

イベントの存続期間の最大差の構成

デフォルトでは、同じエンティティーのイベントは、それぞれの存続期間に関係なく相互に抑止し合います。同じエンティティーについて、今日受け取ったイベントは昨日受け取ったイベントを抑止することができます。これを変更するには、RCA プラグインを通過するイベント間の存続期間の最大差を指定します。存続期間の差がここに指定する値を上回るイベントは、相互に抑止し合うことはできません。

RCA プラグイン用の構成ファイルは RCASchema.cfg 構成ファイルです。このファイルの場所は、`$NCHOME/etc/precision/RCASchema.cfg`です。イベント間の存続期間の最大差の値は、config.defaults テーブルの MaxAgeDifference フィールドに格納されています。

1. `RCASchema.cfg` 構成ファイルを開きます。
2. config.defaults テーブルに対する insert ステートメントを確認します。 デフォルトでは、この insert ステートメントの形式は以下のとおりです。

```
insert into config.defaults
(
  RequeueableEventIds,
  MaxAgeDifference
  HonourManagedStatus
)
values
(
    [
        'NmosPingFail',
        'NmosSnmpPollFail'
    ],
    0
);
```

   デフォルトでは MaxAgeDifference の値は 0 です。これはこのフィーチャーをオフにすることを意味します。
3. このステートメントを変更して、MaxAgeDifference フィールドを目的の値 (分単位) に設定します。

   ヒント: 例えば、MaxAgeDifference フィールドの値を 15 に設定すると、同じエンティティーにおいて存続期間の差が 15 分を上回るイベントが相互に抑止し合うことができないように、システムが構成されます。

関連資料:

278 ページの『config.defaults データベース表』
config.defaults データベース表は、RCA プラグイン・イベント・キューの構成デー

タを保管します。

クロスドメイン・ネットワークでの RCA の考慮事項

Fix Pack 4

クロスドメイン環境では、各ディスカバリー・ドメインの **ncp_g_event** プロセスは、同じディスカバリー・ドメイン内のデバイスに対して RCA を実行します。各ドメイン内では、RCA は、単一のドメインしかない場合と同様に動作します。複数のドメインがクロスドメイン・ディスカバリーを使用して一緒に視覚化されている場合、複数のドメインで根本原因を分析することもできます。

包含抑止

インターフェースは、それが含まれているシャーシと同じドメインに属するため、包含 RCA はクロスドメイン環境の影響を受けません。

接続されているインターフェース

クロスドメイン環境では、ほとんどの接続は同じドメイン内の 2 つのインターフェース間のものであるため、接続抑止は期待通りに機能します。インターフェースが異なるドメインに属している場合、接続抑止はリンクの両端のイベントを関連付けません。

分離抑止

ネットワークの終端にあるデバイス (接続が少ない) は、そのデバイスとポーリング・プログラム・エンティティーの間にあるデバイスで障害が発生すると、ポーリング・プログラム・エンティティーから到達できなくなる (分離される) ことがあります。分離されたすべてのデバイスが同じドメインに属している場合、このような分離されたデバイスでのイベントは抑止されます。ネットワークをパーティション化する際には、分離されたデバイスのグループが同じドメインに属した状態に保たれるようにしてください。

SAE RCA

サービス (例えば、VPN) が 2 つのドメインに属するデバイスで構成されている場合、同じ VPN に対して、各ドメインに 1 つずつ、2 つの SAE イベントが作成されることがあります。

付録 A. デフォルトのポーリング・ポリシー

Network Manager IP Editionには、デフォルトのポーリング・ポリシーのセットが用意されています。 それらのポリシーについて以下にまとめます。

デフォルトの ping ポリシー

Network Manager IP Editionには、ping 操作に関するデフォルトのポーリング・ポリシーが用意されています。

デフォルトの ping ポーリング・ポリシーに関する情報を以下の表にまとめます。

表 52. デフォルトの *ping* ポーリング・ポリシー

ポーリング・ポリシー名	説明
デフォルト・シャーシ ping (Default Chassis Ping)	「デフォルト・シャーシ ping」ポーリング定義は、すべてのネットワーク・デバイスを ping するために使用します。ping 対象のデバイスのタイプは、「デフォルト・シャーシ ping」ポーリング定義のクラス・フィルターによって制限されます。 デフォルトで有効になる唯一のポーリング・ポリシーです。
デフォルト・インターフェース ping	「デフォルト・インターフェース ping」ポーリング定義は、有効な IP アドレスのメイン・ノード内のすべてのインターフェースに対して ping 操作を実行するために使用します。このポリシーは、「デフォルト・インターフェース ping」ポーリング定義を使用して、すべてのネットワーク・デバイスのインターフェースを ping します。ping 対象のインターフェースは、以下に応じて制限されます。 • ポーリング定義内のインターフェース・フィルター。ポーリング定義フィルターを追加して、さらに絞り込むことができます。 • 各インターフェースの管理状況。 TagManagedEntities スティッチャーを変更することにより、変更できます。
エンド・ノード ping	「エンド・ノード ping」ポーリング定義を使用すれば、クラス設定で定義されているすべてのエンド・ノードで ping 操作を実行できます。
ConfirmDeviceDown	さらに高いポーリング頻度、および NmosPingFail イベントが発生したデバイスのみを含んだポリシー・スコープを指定して、デフォルト・シャーシ ping ポーリング定義を使用します。このポリシーは、適応ポーリングのシナリオの一部として使用され、ping ポーリングへの応答に失敗したデバイスに対する高速 ping ポーリングを実行して、実際にダウンしているデバイスを識別するという目的があります。

© Copyright IBM Corp. 2006, 2016

デフォルトのリモート ping ポリシー

Network Manager IP Editionには、リモート ping 操作に関するデフォルトのポーリング・ポリシーが用意されています。これらのポーリングは、SNMP 書き込み操作を使用して、DISMAN-PING-MIB に対するベンダー固有の拡張機能を制御します。

デフォルトのリモート ping ポーリング・ポリシーに関する情報を以下の表にまとめます。

表 53. デフォルトのリモート *ping* ポーリング・ポリシー

ポーリング・ポリシー名	説明
Cisco リモート ping	「Cisco リモート ping」ポーリング定義は、SNMP リモート ping 操作によって、Cisco のプロバイダー・エッジ (PE) デバイスとカスタマー・エッジ (CE) デバイスとの間の MPLS パスの可用性を確認するために使用します。 このポーリング・ポリシーは Cisco デバイスにのみ適用されます。
Juniper リモート ping	「Juniper リモート ping」ポーリング定義を使用すれば、SNMP リモート ping 操作によって、クラス設定で定義されている Juniper の PE デバイスと CE デバイスの間の MPLS パスの可用性を確認できます。 このポーリング・ポリシーは Juniper デバイスにのみ適用されます。

制約事項: Cisco リモート ping、Juniper リモート ping、および汎用しきい値の各ポーリング定義では、ポーリング・データの保管はサポートされません。

デフォルトの SNMP しきい値ポリシー

本製品には、デフォルトの SNMP しきい値ポーリング・ポリシーが用意されています。これらのポーリング・ポリシーは、基本 または 汎用 に分類され、それに加えて、ベンダー固有のものもあります。

しきい値ポリシー

以下の表は、SNMP しきい値ポーリング・ポリシーを示しています。

表 54. 基本 *SNMP* しきい値ポーリング・ポリシー

名前	説明
bgpPeerState	ポーリング定義タイプ: 汎用しきい値。クラス設定およびスコープ設定により定義されるように、bgpPeerState ポーリング定義を使用して、IP 転送機能を持つすべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。

表 54. 基本 SNMP しきい値ポーリング・ポリシー (続き)

名前	説明
ConfirmHighDiscardRate	ポーリング定義タイプ: 基本しきい値。高速 SNMP ポーリングを実現します。クラス設定により定義されるように、HighDiscardRate ポーリング定義を使用して、パケット破棄率 5% のしきい値を違反したインターフェースを 1 つ以上持つすべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。このポリシーは、適応ポーリングのシナリオの一部として使用され、高速ポーリングを実行してデバイス・インターフェースでのしきい値違反を確認するという目的があります。
dot3StatsAlignmentErrors	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、dot3StatsAlignmentErrors ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
frCircuitReceivedBECNs	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、frCircuitReceivedBECNs ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
frCircuitReceivedFECNs	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、frCircuitReceivedFECNs ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
HighDiscardRate	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、HighDiscardRate ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。このポリシーでは、この情報が 30 分ごとにポーリングされます。
ifInDiscards	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、ifInDiscards ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
ifInErrors	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、ifInErrors ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。

表 54. 基本 SNMP しきい値ポーリング・ポリシー (続き)

名前	説明
ifOutDiscards	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、ifOutDiscards ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
ifOutErrors	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、ifOutErrors ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
frCircuitState	ポーリング定義タイプ: 汎用しきい値。クラス設定により定義されるように、frCircuitState ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
isdnLinkUp	ポーリング定義タイプ: 汎用しきい値。クラス設定により定義されるように、isdnLinkUp ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
Fix Pack 5 ospfNbrState	Fix Pack 5 ポーリング定義タイプ: 汎用しきい値。ospfNbrState ポーリング定義を使用して、OSPF ルーターに対してしきい値ポーリングを実行します。
rebootDetection	ポーリング定義タイプ: 汎用しきい値。クラス設定により定義されるように、rebootDetection ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
snmpInBandwidth	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、snmpInBandwidth ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。
snmpOutBandwidth	ポーリング定義タイプ: 基本しきい値。クラス設定により定義されるように、snmpOutBandwidth ポーリング定義を使用して、すべてのネットワーク・デバイスに対してしきい値ポーリングを実行します。

Foundry SNMP しきい値ポーリング・ポリシー

以下の表は、Foundry デバイスに提供される SNMP しきい値ポーリング・ポリシーを示しています。

表 55. *Foundry* デバイスの *SNMP* しきい値ポーリング・ポリシー

名前	説明
snChasActualTemperature	クラス設定により定義されるように、snChasActualTemperature ポーリング定義を使用して、すべての Foundry デバイスに対してしきい値ポーリングを実行します。
snChasFanOperStatus	クラス設定により定義されるように、snChasFanOperStatus ポーリング定義を使用して、すべての Foundry デバイスに対してしきい値ポーリングを実行します。
snChasPwrSupplyOperStatus	クラス設定により定義されるように、snChasPwrSupplyOperStatus ポーリング定義を使用して、すべての Foundry デバイスに対してしきい値ポーリングを実行します。

Cisco SNMP しきい値ポリシー

以下の表は、Cisco デバイスに提供される SNMP しきい値ポーリング・ポリシーを示しています。

表 56. *Cisco* デバイスの *SNMP* しきい値ポーリング・ポリシー

名前	説明
bufferPoll	クラス設定により定義されるように、bufferPoll ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。
ciscoEnvMonFanState	クラス設定により定義されるように、ciscoEnvMonFanState ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。
ciscoEnvMonSupplyState	クラス設定により定義されるように、ciscoEnvMonSupplyState ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。
ciscoEnvMonTemperature 状態	クラス設定により定義されるように、ciscoEnvMonTemperatureState ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。
memoryPoll	クラス設定により定義されるように、ciscoMemoryPool ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。
cpuBusyPoll	クラス設定により定義されるように、cpuBusyPoll ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。
locIfInCrcErrors	クラス設定により定義されるように、locIfInCrcErrors ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。
memoryPoll	クラス設定により定義されるように、memoryPoll ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。

表 56. *Cisco* デバイスの *SNMP* しきい値ポーリング・ポリシー *(*続き*)*

名前	説明
sysTrafficPoll	クラス設定により定義されるように、sysTrafficPoll ポーリング定義を使用して、すべての Cisco デバイスに対してしきい値ポーリングを実行します。

デフォルトの SNMP リンク状態ポリシー

デフォルトの SNMP リンク状態ポーリング・ポリシーは、「SNMP リンク状態」ポーリング定義を使用して、クラス設定で定義されているすべてのネットワーク・デバイスの管理状況と操作状況を確認します。インターフェースの状況に変化があると、イベントが生成されます。

レポート作成で使用されるポーリング・ポリシー

レポート作成に特化して定義されたポーリング・ポリシーはありません。ここでの情報に基づいて、既存のポーリング・ポリシーのいずれがレポート作成で使用されるかを理解してください。

一部のレポートには、特定のポーリング・ポリシーを有効にする必要があります。これらのレポートおよびポリシーを、表 57に示します。

表 57. レポート作成で使用されるポーリング・ポリシー

レポート	有効にする必要があるポーリング・ポリシー
デバイス出口トラフィックの要約	ifOutError ifOutDiscards snmpOutBandwidth
デバイス入り口トラフィックの要約	ifInError ifInDiscards snmpInBandwidth
ルーターの正常性の要約	ciscoCPUTotal5Min ciscoMemoryPctgUsage デフォルトのシャーシ・ポーリング

付録 B. デフォルトのポーリング定義

Network Manager IP Editionには、一般的なポーリング要件に対応したさまざまなデフォルトのポーリング定義が用意されています。

Network Manager IP Editionに用意されているデフォルトのポーリング定義を以下の表にまとめます。

デフォルトの SNMP ポーリング定義

bgpPeerState

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、BGP ピア状態チェックが定義されています。Border Gateway Patrol (BGP) ピアが非確立状態に変わると、アラートが生成されます。このポーリング定義は、bgpPeerState MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

パス: `iso/org/dod/mgmt/mib-2/bgpPeerTable/bgpPeerEntry/bgpPeerState`

MIB OID: 1.3.6.1.2.1.15.3.1.2

bufferPoll

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、バッファー消費チェックが定義されています。空きバッファー・エレメントの数が 100 を下回ると、アラートが生成されます。このポーリング定義は、bufferElFree MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: `iso/org/dod/private/enterprises/cisco/local/bufferElFree`

MIB OID: 1.3.6.1.4.1.9.2.1.9

ciscoCPUTotal5min

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、CPU 使用量チェックが定義されています。cpmCPUTotal5min Cisco MIB 変数の値が 80% を超えると、アラートが生成されます。このポーリング定義は、直前の 5 分間の全体的な CPU ビジー率を示す cpmCPUTotal5min MIB 変数に対してポーリングを実行します。このオブジェクトによって、OLD-CISCO-SYSTEM-MIB の avgBusy5 オブジェクトは非推奨となりました。cpmCPUTotal5min MIB 変数のパスと OID は以下のとおりです。

MIB パス: `iso/org/dod/private/enterprises/cisco/local/cpmCPUTotal5min`

MIB OID: 1.3.6.1.4.1.9.9.109.1.1.1.1.5

ciscoEnvMonFanState

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、Cisco デバイスのファン状況チェックが定義されています。状況が 1 (正常) 以外の値に変わると、アラートが生成されます。このポーリング定義は、ciscoEnvMonFanState MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

© Copyright IBM Corp. 2006, 2016

MIB パス: iso/org/dod/mgmt/mib-2/private/enterprises/cisco/ciscoMgmt/ciscoEnvMonMIB/ciscoEnvMonObjects/ciscoEnvMonFanStatusTable/ciscoEnvMonFanStatusEntry/ciscoEnvMonFanState

MIB OID: 1.3.6.1.4.1.9.9.13.1.4.1.3

ciscoEnvMonSupplyState

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、Cisco デバイスの電源機構状況チェックが定義されています。状況が 1 (正常) 以外の値に変わると、アラートが生成されます。このポーリング定義は、ciscoEnvMonSupplyState MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/private/enterprises/cisco/ciscoMgmt/ciscoEnvMonMIB/ciscoEnvMonObjects/ciscoEnvMonSupplyStatusTable/ciscoEnvMonSupplyStatusEntry/ciscoEnvMonSupplyState

MIB OID: 1.3.6.1.4.1.9.9.13.1.5.1.3

ciscoEnvMonTemperatureState

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、Cisco デバイスのファン温度状況チェックが定義されています。状況が 1 (正常) 以外の値に変わると、アラートが生成されます。このポーリング定義は、ciscoEnvMonTemperatureState MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/private/enterprises/cisco/ciscoMgmt/ciscoEnvMonMIB/ciscoEnvMonObjects/ciscoEnvMonTemperatureStatusTable/ciscoEnvMonTemperateureStatusEntry/ciscoEnvMonTemperatureState

MIB OID: 1.3.6.1.4.1.9.9.13.1.3.1.6

Cisco リモート ping

ポーリング定義タイプ: Cisco リモート ping。このポーリング定義では、Cisco 固有の MIB を使用するリモート ping 操作が定義されています。

cpuBusyPoll

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、CPU 使用量チェックが定義されています。avgBusy5 Cisco MIB 変数の値が 80% を超えると、アラートが生成されます。このポーリング定義は、avgBusy5 MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/private/enterprises/cisco/local/avgBusy5

MIB OID: 1.3.6.1.4.1.9.2.1.58

制約事項**:** 最近の Cisco ルーターの中には、aveBusy5 MIB 変数をサポートしないものがあります。このようなルーターに対しては、ciscoCPUTotal5min ポーリング定義を使用します。

HighDiscardRate

ポーリング定義タイプ: 基本しきい値。デバイス上の 1 つ以上のインター

フェースでパケット破棄率が 5% を超えると、アラートが生成されます。このポーリング定義は、以下の MIB 変数に対してポーリングを実行します。

ifInDiscards

MIB パス: iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifInDiscards

MIB OID: 1.3.6.1.2.1.2.2.1.13

ifInNUcastPkts

MIB パス: iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifInNUcastPkts

MIB OID: 1.3.6.1.2.1.2.2.1.12

ifInUcastPkts

MIB パス: iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifInUcastPkts

MIB OID: 1.3.6.1.2.1.2.2.1.11

dot3StatsAlignmentErrors

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、位置合わせのエラー率チェックが定義されています。エラー率が 1 秒当たり 0 個を上回ると、アラートが生成されます。このポーリング定義は、dot3StatsAlignmentErrors MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/transmission/dot3/dot3StatsTable/dot3StatusEntry/dot3StatsAlignmentErrors

MIB OID: 1.3.6.1.2.1.10.7.2.1.2

frCircuitReceivedBECNs

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、データ・リンク接続 ID (DLCI) のフレーム・リレー逆方向輻輳 (ふくそう) チェックが定義されています。 DLCI の逆方向輻輳通知を受信すると、アラートが生成されます。このポーリング定義は、frCircuitReceivedBECNs MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/transmission/frameRelayDTE/frCircuitTable/frCircuitEntry/frCircuitReceivedBECNs

MIB OID: 1.3.6.1.2.1.10.32.2.1.5

frCircuitReceivedFECNs

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、DLCI のフレーム・リレー順方向輻輳チェックが定義されています。DLCI の順方向輻輳通知を受信すると、アラートが生成されます。このポーリング定義は、frCircuitReceivedFECNs MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/transmission/frameRelayDTE/frCircuitTable/frCircuitEntry/frCircuitReceivedFECNs

MIB OID: 1.3.6.1.2.1.10.32.2.1.4

frCircuitState

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、フレーム・リレー回線状態チェックが定義されています。回線が非アクティブになると、アラートが生成されます。始動時に非アクティブになっている回線に関するアラートが生成されるのを回避するために、この定義では、回線が非アクティブになったかどうかをチェックします。このポーリング定義は、frCircuitState MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: `iso/org/dod/mgmt/mib-2/transmission/frameRelayDTE/frCircuitTable/frCircuitEntry/frCircuitState`

MIB OID: 1.3.6.1.2.1.10.32.2.1.3

ifInDiscards

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、インバウンド廃棄率チェックが定義されています。比率が 1 秒当たり 0 個を上回ると、アラートが生成されます。このポーリング定義は、ifInDiscards MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifInDiscards`

MIB OID: 1.3.6.1.2.1.2.2.1.13

ifInErrors

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、インバウンド・インターフェース・エラー率チェックが定義されています。比率が 1 秒当たり 0 個を上回ると、アラートが生成されます。このポーリング定義は、ifInErrors MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifInErrors`

MIB OID: 1.3.6.1.2.1.2.2.1.14

ifOutDiscards

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、アウトバウンド廃棄率チェックが定義されています。比率が 1 秒当たり 0 個を上回ると、アラートが生成されます。このポーリング定義は、ifOutDiscards MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifOutDiscards`

MIB OID: 1.3.6.1.2.1.2.2.1.19

ifOutErrors

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、アウトバウンド・インターフェース・エラー率チェックが定義されています。比率が 1 秒当たり 0 個を上回ると、アラートが生成されます。このポーリング定義は、ifOutErrors MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifOutErrors

MIB OID: 1.3.6.1.2.1.2.2.1.20

isdnLinkUp

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、ISDN リンク状態チェックが定義されています。ISDN リンクがアクティブになると、アラートが生成されます。 ISDN リンクがアクティブになるということは、対応する 1 次リンクがダウンしたことを意味する場合があります。このポーリング定義は、以下の MIB 変数に対してポーリングを実行します。

MIB パス: iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifOperStatus

MIB OID: 1.3.6.1.2.1.2.2.1.8

MIB パス: iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifAdminStatus

MIB OID: 1.3.6.1.2.1.2.2.1.7

Juniper リモート ping

ポーリング定義タイプ: Juniper リモート ping。このポーリング定義では、Juniper 固有の MIB を使用するリモート ping 操作が定義されています。

locIfInCrcErrors

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、インバウンド・サイクル冗長度チェックサム (CRC)/位置合わせエラー・チェックが定義されています。インバウンド CRC 位置合わせエラーが発生すると、アラートが生成されます。このポーリング定義は、locIfInCRC MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/private/enterprises/cisco/local/linterfaces/lifTable/lifEntry/locIfInCRC

MIB OID: 1.3.6.1.4.1.9.2.2.1.1.12

memoryPoll

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、メモリー消費チェックが定義されています。空きメモリー量が 100 バイトを下回ると、アラートが生成されます。このポーリング定義は、freeMem MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/private/enterprises/cisco/local/lsystem/freeMem

MIB OID: 1.3.6.1.4.1.9.2.1.8

Fix Pack 5 **ospfNbrState**

ポーリング定義タイプ: 汎用しきい値。このポーリング定義は、OSPF 隣接情報チェックを定義し、OSPF ルーターが full 以外の状態の場合にアラートを生成します。 OSPF ルーターの状態が down、attempt、init、twoWay、exchangeStart、exchange、または loading の場合は、隣接情報の

形成時の問題を示します。 このポーリング定義は、ospfNbrState MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/internet/mgmt/mib-2/ospf/ospfNbrTable/ospfNbrEntry/ospfNbrState

MIB OID: 1.3.6.1.2.1.14.10.1.6

rebootDetection

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、デバイス・リブート検出チェックが定義されています。デバイスがリブートされると、アラートが生成されます。デバイスのリブートの理由としては、デバイスの SNMP サブシステムのリセットが考えられます。

ヒント: システムのアップタイムをモニターするには、sysUpTime MIB 変数を hrSystemUptime MIB 変数に変更します。hrSystemUptime MIB 変数は、HOSTRES-MIB がデバイスでサポートされている場合にのみ使用できます。
このポーリング定義は、sysUpTime MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/system/sysUpTime

MIB OID: 1.3.6.1.2.1.1.3

snChasActualTemperature

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、Foundry デバイスの温度チェックが定義されています。シャーシの実際の温度が、温度レベルに関する警告として設定されている値を超えると、アラートが生成されます。このポーリング定義は、以下の MIB 変数に対してポーリングを実行します。

snChasActualTemperature

MIB パス: iso/org/dod/mgmt/mib-2/private/enterprises/foundry/foundryProducts/switch/snChassis/snChassGen/snChasActualTemperature

MIB OID: 1.3.6.1.4.1.1991.1.1.1.1.18

snChasWarningTemperature

MIB パス: iso/org/dod/mgmt/mib-2/private/enterprises/foundry/foundryProducts/switch/snChassis/snChassGen/snChasWarningTemperature

MIB OID: 1.3.6.1.4.1.1991.1.1.1.1.19

snChasFanOperStatus

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、Foundry デバイスのファン状況チェックが定義されています。ファン状況が 2 (正常) から 3 (障害) に変わると、アラートが生成されます。このポーリング定義は、snChasFanOperStatus MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: iso/org/dod/mgmt/mib-2/private/enterprises/foundry/foundryProducts/switch/snChassis/snChasFan/snChasFanTable/snChasFanEntry/snChasFanOperStatus

MIB OID: 1.3.6.1.4.1.1991.1.1.1.3.1.1.3

snChasPwrSupplyOperStatus

ポーリング定義タイプ: 汎用しきい値。このポーリング定義では、Foundry デバイスの電源機構状況チェックが定義されています。電源機構状況が 2 (正常) から 3 (障害) に変わると、アラートが生成されます。このポーリング定義は、snChasPwrSupplyOperStatus MIB 変数に対してポーリングを実行します。その MIB 変数のパスと OID は以下のとおりです。

MIB パス: `iso/org/dod/mgmt/mib-2/private/enterprises/foundry/foundryProducts/switch/snChassis/snChasPwr/snChasPwrSupplyTable/snChasPwrSupplyEntry/snChasPwrSupplyOperStatus`

MIB OID: 1.3.6.1.4.1.1991.1.1.1.2.1.1.3

SNMP リンク状態

ポーリング定義タイプ: SNMP リンク状態。このポーリング定義では、管理状況/操作状況チェックが定義されています。管理状況と操作状況の間に不一致があると、アラートが生成されます。このポーリング定義は、以下の MIB 変数に対してポーリングを実行します。

ifAdminStatus

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifAdminStatus`

MIB OID: 1.3.6.1.2.1.2.2.1.7

ifOperStatus

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifOperStatus`

MIB OID: 1.3.6.1.2.1.2.2.1.8

snmpInBandwidth

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、着信帯域幅使用率チェックが定義されています。着信帯域幅使用率が 40% を超えると、アラートが生成されます。このポーリング定義は、以下の MIB 変数に対してポーリングを実行します。

ifInOctets

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifInOctets`

MIB OID: 1.3.6.1.2.1.2.2.1.10

ifSpeed

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifSpeed`

MIB OID: 1.3.6.1.2.1.2.2.1.5

snmpOutBandwidth

ポーリング定義タイプ: 基本しきい値。このポーリング定義では、出力 SNMP 帯域幅使用率チェックが定義されています。インターフェースに対して出力 SNMP 帯域幅使用率が 40% を超えると、アラートが生成されます。このポーリング定義は、以下の MIB 変数に対してポーリングを実行します。

ifOutOctets

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifOutOctets`

MIB OID: 1.3.6.1.2.1.2.2.1.16

ifSpeed

MIB パス: `iso/org/dod/mgmt/mib-2/interfaces/ifTable/ifEntry/ifSpeed`

MIB OID: 1.3.6.1.2.1.2.2.1.5

sysUpTime

ポーリング定義タイプ: 基本しきい値。このポーリングでは、sysUpTime MIB 変数の値を取得します。この MIB 変数のパスと OID は以下のとおりです。

MIB パス: `iso/org/dod/mgmt/mib-2/system/sysUpTime`

MIB OID: 1.3.6.1.2.1.1.3

デフォルトの ping ポーリング定義

デフォルト・シャーシ **ping (Default Chassis Ping)**

ポーリング定義タイプ: シャーシ ping。このポーリング定義では、メイン・ノード・デバイスに対する ping 操作が定義されています。デバイスのメイン・ノード IP アドレスに ICMP パケットが送信されます。

デフォルト・インターフェース **ping**

ポーリング定義タイプ: インターフェース ping。このポーリング定義では、デバイスに含まれているインターフェースに対する ping 操作が定義されています。各インターフェースの IP アドレスに ICMP パケットが送信されます。

エンド・ノード **ping**

ポーリング定義タイプ: シャーシ ping。このポーリング定義では、プリンターやワークステーションなどのエンド・ノードに対する ping 操作が定義されています。各エンド・ノードの IP アドレスに ICMP パケットが送信されます。

これらの ping ポーリング定義のそれぞれについて収集できる ping メトリックは、応答時間とパケット・ロスです。

付録 C. トリガーしきい値とクリアのしきい値の例

しきい値の数式の例を参考にしながら、汎用しきい値ポーリング定義のクリアのしきい値とトリガーしきい値を設定できます。

トリガーしきい値の例

以下のような場合にイベントを生成するための例を取り上げます。

- avgBusy5 MIB 変数の現行値が avgBusy6 MIB 変数の値以上になっている場合

そのしきい値を作成するには、「ポーリング定義エディター」の「トリガーしきい値」タブで以下の情報を指定します。

1. 「基本」を選択します。
2. 最初のしきい値を指定します。
   a. 「現在」を選択します。
   b. 「**MIB** オブジェクトの追加」 をクリックします。
   c. 「**MIB** ツリー」を以下のパスに展開します。

      `iso/org/dod/internet/private/enterprises/cisco/local/lsystem/avgBusy5`

      「挿入」をクリックします。
   d. 条件として「>=」を選択します。
   e. 「現在」を選択します。
   f. 「**MIB** オブジェクトの追加」 をクリックします。
   g. 「MIB ツリー」を以下のパスに展開します。

      `iso/org/dod/internet/private/enterprises/cisco/local/lsystem/avgBusy6`

      「挿入」をクリックします。
3. イベントが生成されたときにアクティブ・イベント・リストに表示されるメッセージを指定します。
   a. 「イベント記述」フィールドに `CPU usage high (avgBusy5=` と入力します。
   b. 「**MIB** オブジェクトの追加」 をクリックします。
   c. 「**MIB** ツリー」を以下のパスに展開します。

      `iso/org/dod/internet/private/enterprises/cisco/local/lsystem/avgBusy5`
   d. 「現行 **SNMP** 値」を選択して、「挿入」をクリックします。
   e. `>=` と入力します。
   f. 「**MIB** オブジェクトの追加」 をクリックします。
   g. 「**MIB** ツリー」を以下のパスに展開します。

      `iso/org/dod/internet/private/enterprises/cisco/local/lsystem/avgBusy6`

© Copyright IBM Corp. 2006, 2016

h.) と入力します。

アクティブ・イベント・リストの記述は、以下のようになります。

```
CPU usage high (avgBusy5=eval(text,"SNMP.VALUE.avgBusy5")>=eval
(text,"SNMP.VALUE.avgBusy6"))
```

クリアのしきい値の例

以下のような場合にクリア・イベントを生成するための例を取り上げます。

- avgBusy5 MIB 変数の値が 80 未満の場合

そのしきい値を作成するには、「ポーリング定義エディター」の「クリアのしきい値」タブで以下の情報を指定します。

1. 「基本」を選択します。
2. しきい値を指定します。
   a. 「現在」を選択します。
   b. 「**MIB** オブジェクトの追加」 をクリックします。
   c. 「**MIB** ツリー」を以下のパスに展開します。

      ```
      iso/org/dod/internet/private/enterprises/cisco/local/lsystem/avgBusy5
      ```

      「挿入」をクリックします。
   d. 条件として「<=」を選択します。
   e. 「**literal**」を選択します。
   f. 80 と入力します。
3. イベントが生成されたときにアクティブ・イベント・リストに表示されるメッセージを指定します。
   a. 「イベント記述」フィールドに CPU usage high (avgBusy5= と入力します。
   b. 「**MIB** オブジェクトの追加」 をクリックします。
   c. 「**MIB** ツリー」を以下のパスに展開します。

      ```
      iso/org/dod/internet/private/enterprises/cisco/local/lsystem/avgBusy5
      ```
   d. 「現行 **SNMP** 値」を選択して、「挿入」をクリックします。
   e. <= と入力します。
   f. 80 と入力します。
   g.) と入力します。

   アクティブ・イベント・リストの記述は、以下のようになります。

   ```
   CPU usage high (avgBusy5=eval(text,"SNMP.VALUE.avgBusy5")<=80)
   ```

付録 D. ポーリング定義の式の構文

基本しきい値ポーリング定義と汎用しきい値ポーリング定義で使用するために複雑なしきい値の式を作成する方法を理解するには、この情報を使用してください。

関連資料:

54 ページの『基本しきい値の式の例』
この基本しきい値の式の例を使用して、複雑な基本しきい値の式を構成する方法を理解してください。

54 ページの『汎用しきい値の式の例』
この汎用しきい値の式の例を使用して、複雑な汎用しきい値の式を構成する方法を理解してください。

しきい値の式での eval ステートメントの構文

ここでは、基本しきい値ポーリング定義と汎用しきい値ポーリング定義内で、eval ステートメントを使用して複雑なしきい値の式を作成する方法について説明します。

SNMP 変数に対する eval ステートメントの構文

eval ステートメントを使用して SNMP 変数を評価することができます。

以下の各例は、eval ステートメントを使用して SNMP 変数を評価する方法を示しています。

サンプル: SNMP 値の評価

以下の例は、SNMP 変数 sysName の値を返します。

```
eval(text, '&SNMP.VALUE.sysName')
```

サンプル: SNMP 索引の評価

以下の例は、変数 ipRouteNextHop に対する SNMP 要求の索引の値を返します。テーブル・ポーリングでは、テーブル・リスト内の索引ごとにこの変数が評価されます。

```
eval(text, '&SNMP.INDEX.ipRouteNextHop')
```

サンプル: 以前に取得された SNMP 値の評価

以下の例は、このポーリングが前回実行されたときに取得された SNMP 変数 sysName の値を返します。

```
eval(text, '&SNMP.VALUE.OLD.sysName')
```

サンプル: 以前の SNMP 索引の評価

以下の例は、このポーリングが前回実行されたときに取得された、変数 ipRouteNextHop に対する SNMP 要求の索引の値を返します。テーブル・ポーリン

© Copyright IBM Corp. 2006, 2016

グでは、テーブル・リスト内の索引ごとにこの変数が評価されます。前の索引は、新しい索引と同じである可能性が高いことに注意してください。

```
eval(text, '&SNMP.INDEX.OLD.ipRouteNextHop')
```

ネットワーク・エンティティー変数に対する eval ステートメントの構文

eval ステートメントを使用して、エンティティー ID やエンティティー名の値などのネットワーク・エンティティー変数を評価することができます。

ENTITY キーワードをしきい値の式または説明で使用して、ネットワーク・エンティティー変数の値を評価することができます。以下の例は、eval ステートメントで ENTITY キーワードを使用してネットワーク・エンティティー変数を評価する方法を示しています。

サンプル: 収容シャーシの entityName の値の評価

以下の例は、しきい値の式またはしきい値の説明で使用され、モニターされているエンティティーを含んでいるシャーシに対応する entityName の値を評価する方法を示しています。

eval(text, '&ENTITY.MAINNODEENTITYNAME') 上のインターフェースがダウンしています

ENTITY キーワードの属性

ENTITY キーワードの有効な属性を、以下の表に示します。この表に示している NCIM データベース・フィールドについては、「*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンス」を参照してください。

表 *58. ENTITY* キーワードの属性

属性	説明
ENTITYID	モニター対象エンティティーの entity.entityId フィールドの値。これはインターフェースまたはシャーシのいずれかです。
ENTITYNAME	モニター対象エンティティーの entity.entityName フィールドの値。これはインターフェースまたはシャーシのいずれかです。
ENTITYTYPE	モニター対象エンティティーの entity.entityType フィールドの値。これはインターフェースまたはシャーシのいずれかです。
ENTITYCLASS	モニター対象エンティティーの entity.className フィールドの値。これはインターフェースまたはシャーシのいずれかです。
ACCESSIPADDRESS	ポーリングのタイプに応じて、interface.accessIPAddress または chassis.accessIPAddress フィールドの値。
IFINDEX	インターフェース・ポーリングでの、モニター対象エンティティーの interface.ifIndex フィールドの値。
IFTYPESTRING	インターフェース・ポーリングでの、モニター対象エンティティーの interface.ifTypeString フィールドの値。
IFNAME	インターフェース・ポーリングでの、モニター対象エンティティーの interface.ifName フィールドの値。
IFDESCR	インターフェース・ポーリングでの、モニター対象エンティティーの interface.ifDescr フィールドの値。

表 58. *ENTITY* キーワードの属性 (続き)

属性	説明
IFALIAS	インターフェース・ポーリングでの、モニター対象エンティティーの interface.ifAlias フィールドの値。
INSTANCESTR	インターフェース・ポーリングでの、モニター対象エンティティーの interface.instanceStr フィールドのストリング表現。
ENTITYMANAGED	モニター対象エンティティーが管理状態にあるかどうかを示します。これは、当該エンティティーに対して managedStatus.status フィールドが存在しないか、またはゼロであるかで判別されます。
CHASSISMANAGED	モニター対象エンティティーを含んでいるシャーシが管理状態にあるかどうかを示します。これは、当該シャーシに対して managedStatus.status フィールドが存在しないか、またはゼロであるかで判別されます。
MAINNODEADDRESS	モニター対象エンティティーを含んでいるシャーシの accessIPAddress の値。
MAINNODEENTITYNAME	モニター対象のエンティティーを含んでいるシャーシに対応するエンティティー・レコードの entityName フィールドの値。
MAINNODEENTITYID	モニター対象エンティティーを含んでいるシャーシの chassis.entityId フィールドの値。

ポーリング・ポリシー変数に対する eval ステートメントの構文

eval ステートメントを使用して、ポリシーの名前、このポーリング・ポリシーが見つかったドメインの ID などのポーリング・ポリシー変数を評価できます。

POLICY キーワードをしきい値の式または説明で使用して、ポーリング・ポリシー変数の値を評価することができます。以下の例は、eval ステートメントで POLICY キーワードを使用してポーリング・ポリシー変数を評価する方法を示しています。

サンプル: ポーリング・ポリシー名および関連するドメイン ID の値の評価

以下の例は、しきい値の式またはしきい値の説明で使用され、ポーリング・ポリシーの名前、およびこのポーリング・ポリシーが存在するドメインの ID の値を評価する方法を示しています。

eval(text, '&POLICY.POLICYNAME') ポリシーは、
ドメイン番号 **eval(text, '&POLICY.DOMAINMGRID)** 内のエンティティーをポーリングします

POLICY キーワードの属性

POLICY キーワードの有効な属性を、以下の表に示します。この表に示している NCIM データベース・フィールドについては、「*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンス」を参照してください。

表 59. *POLICY* キーワードの属性

属性	説明
POLICYID	このポーリング・ポリシーの固有の整数 ID。
DOMAINMGRID	NCIM domainMgr テーブルを参照している外部キー。モニター対象エンティティーのドメインの ID を指定します。

表 59. POLICY キーワードの属性 (続き)

属性	説明
POLICYNAME	このポーリング・ポリシーの名前

ポーリング定義変数に対する eval ステートメントの構文

eval ステートメントを使用して、ポーリング定義の名前や、ポーリング定義を使用するポリシーによって生成される障害イベントの重大度などのポーリング定義変数を評価することができます。

POLL キーワードをしきい値の式または説明で使用して、ポーリング定義変数の値を評価することができます。以下の例は、 eval ステートメントで POLL キーワードを使用してポーリング定義変数を評価する方法を示しています。

サンプル: ポーリング定義名および関連するイベント重大度の値の評価

以下の例は、しきい値の式またはしきい値の説明で使用され、ポーリング定義の名前、およびこのポーリング定義を使用するポーリング・ポリシーによって生成されるイベントの重大度の値を評価する方法を示しています。

eval(text, '&POLL.TEMPLATENAME') ポーリング定義を使用するポーリング・ポリシーによって
重大度 **eval(text, '&POLL.EVENTSEVERITY')** のイベントが生成されます

POLL キーワードの属性

POLL キーワードの有効な属性を、以下の表に示します。

表 60. POLL キーワードの属性

属性	説明
TEMPLATEID	このポーリング定義の固有 ID。
TEMPLATENAME	このポーリング定義の名前。
TEMPLATETYPE	ポーリング定義のタイプ。この値は、ユーザーが新規のポーリング定義を作成するときに表示されるリストから得られます。
EVENTNAME	このポーリング定義を使用するポーリング・ポリシーによって生成されるイベントに使用されるテキスト ID。このテキストは、Tivoli Netcool/OMNIbus (nco_p_ncpmonitor) のプローブのルール・ファイルによって変更されない限り、alerts.status テーブルに EventId フィールドとして書き込まれます。
EVENTSEVERITY	このポーリング定義を使用するポーリング・ポリシーによって生成される障害イベントの重大度。このテキストは、Tivoli Netcool/OMNIbus (nco_p_ncpmonitor) のプローブのルール・ファイルによって変更されない限り、alerts.status テーブルに Severity フィールドとして書き込まれます。
POLLINTERVAL	このポーリングのスコープ内にある各エンティティーがポーリングされる間隔 (秒単位)。

しきい値の式での演算子

ここでは、基本しきい値ポーリング定義と汎用しきい値ポーリング定義内で、しきい値の式を作成するために使用する演算子について説明します。

以下の表に、しきい値の式で使用できる演算子をリストします。

表 *61.* しきい値の式での演算子

演算子	例
加算	(1 + 2)
減算	(4 - 2)
乗算	(5 * 3)
除算	(10 / 2)
係数	(8 % 3)
累乗	(10 POW 3)
対数	(Ln 5)
IP から Long へのデータ型変換	(IpToLong("1.2.3.4"))
ビット単位 AND	(5 & 3) **注:** ビット単位演算は整数値にのみ適用できます。
ビット単位	(5 | 3)
ビット単位排他 OR	(5 ^ 3)
ブール OR	((eval(int, '&SNMP.VALUE.ifSpeed') > 10000) OR (eval(int, '&SNMP.VALUE.ifSpeed') < 100))
ブール AND	((eval(int, '&SNMP.VALUE.ifSpeed') > 10000) AND (eval(int, '&SNMP.VALUE.ifOperStatus') !=2))
ブール NOT	(NOT((eval(int,'&SNMP.VALUE.ifOperStatus') = 1))
等しい	(eval(int, '&SNMP.VALUE.ifOperStatus') = 1)
等しくない	(eval(int, '&SNMP.VALUE.ifOperStatus') != 1)
より小	(eval(int, '&SNMP.VALUE.ifSpeed') < 100)
より大	(eval(int, '&SNMP.VALUE.ifSpeed') >100)
以下	(eval(int, '&SNMP.VALUE.ifSpeed') <= 100)
以上	(eval(int, '&SNMP.VALUE.ifSpeed') >= 100)
類似	(eval(text, '&ENTITY.IFDESCR') LIKE 'Gigabit.*')
除外	(eval(text, '&ENTITY.IFDESCR') NOT LIKE 'Loopback.*')

付録 E. Tivoli Netcool/OMNIbus のプローブ の構成

Tivoli Netcool/OMNIbus のプローブ (**nco_p_ncpmonitor**) は、Network Manager のポーリングおよびプロセスによって生成されたイベントを取得して処理し、ObjectServer に転送します。

Tivoli Netcool/OMNIbus のプローブは、`$NCHOME/probes/arch` ディレクトリーにインストールされます。この *arch* は、オペレーティング・システムのディレクトリーを表します。プローブは、以下に示すそのプローブの構成ファイルを使用して構成することができます。

- プロパティー・ファイル: `nco_p_ncpmonitor.props`
- ルール・ファイル: `nco_p_ncpmonitor.rules`

注: プローブの実行可能ファイル (または **nco_p_ncpmonitor** コマンド) も、`$NCHOME/probes/arch` ディレクトリーにインストールされます。ただしプローブは、デフォルトでドメイン・プロセス・コントローラー CTRL の下で稼働するように構成され、**nco_p_ncpmonitor** コマンドはトラブルシューティングのみを目的として手動で実行する必要があります。

Network Manager で生成されたイベントは、ドメイン固有です。Network Manager がフェイルオーバー・モードで稼働しているときは、プローブがデフォルトで仮想ドメイン名を使用します (その名前が `$NCHOME/etc/precision/ConfigItnm.cfg` ファイルで構成されている場合)。

プローブの概念について詳しくは、Tivoli Netcool/OMNIbus インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.tivoli.nam.doc/welcome_ob.htm) の「*IBM Tivoli Netcool/OMNIbus* プローブとゲートウェイ・ガイド」を参照してください。

nco_p_ncpmonitor.props ファイルについて

`$NCHOME/probes/arch/nco_p_ncpmonitor.props` ファイルは、Tivoli Netcool/OMNIbus のプローブが稼働する環境を定義します。

このプロパティー・ファイルは、コロンで区切られた名前と値のペアから構成されます。デフォルトのプロパティー・ファイルには、プローブがサポートするプロパティーのサブセットがリストされます。これらのプロパティーは、行の先頭に番号記号 (#) を付けてコメント化されています。適切であれば、稼働中の Tivoli Netcool/OMNIbus のバージョンに適用される共通のプローブ・プロパティーの標準セットを、Tivoli Netcool/OMNIbus のプローブに指定することができます。

プロパティーのデフォルト値を変更する場合は、必要な各プロパティーの名前と値の行をファイルの最後に追加することが推奨されています。プロパティーを指定する場合は、行がアンコメントされていることを確認してから、必要に応じて値を変更します。ストリング値は引用符で囲む必要があります。その他の値は引用符で囲む必要はありません。例えば、次のようになります。

© Copyright IBM Corp. 2006, 2016

```
Server            :    "VIRTUAL"
RulesFile         :    "$NCHOME/probes/solaris2/nco_p_ncpmonitor.rules"
Buffering         :    1
BufferSize        :    15
```

トラブルシューティングの目的で、コマンド行から関連するコマンド行オプションを指定した **nco_p_ncpmonitor** コマンドを実行してプローブのプロパティーを構成することもできます。

プローブ共通のプロパティーについては、Tivoli Netcool/OMNIbus インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.tivoli.nam.doc/welcome_ob.htm) の「*IBM Tivoli Netcool/OMNIbus* プローブとゲートウェイ・ガイド」を参照してください。

nco_p_ncpmonitor.rules ファイルについて

$NCHOME/probes/*arch*/nco_p_ncpmonitor.rules ファイルでは、Tivoli Netcool/OMNIbus のプローブが Network Manager のイベント・データをどのように処理して意味のある Tivoli Netcool/OMNIbus イベントを作成するかが定義されています。

nco_p_ncpmonitor.rules 構成のリファレンス

ルール・ファイルは、Network Manager のイベント・データを ObjectServer のフィールドにマップするもので、プローブの動作をカスタマイズするために使用できます。ルール・ファイルを構成するには、Tivoli Netcool/OMNIbus のプローブ・ルールの構文に関する知識が必要です。

プローブでは、トークンとエレメントを使用し、さらにルールを適用して、Network Manager イベント・ソースのデータを ObjectServer で認識できるフォーマットに変換します。未加工のイベント・ソース・データはトークンに変換され、さらにエレメントへと解析されます。ルール・ファイルは、これらのエレメントに対して条件付き処理を実行し、ObjectServer の alerts.status のフィールドにマップするために使用されます。ルール・ファイルでは、エレメントは $ 記号で識別され、alerts.status のフィールドは @ 記号で識別されます。ルール・ファイル構成では、以下のサンプル・コードに示すように、エレメントがフィールドにマップされます。

```
@Summary=$Description
```

この例では、@Summary が alerts.status のフィールドを表し、$Description が Network Manager の入力フィールドを表します。

エンティティーの追加データを保管するために、Network Manager の ExtraInfo フィールドがネストされたフィールドとともに使用されている場合 (例えば ExtraInfo->ifIndex)、ルール・ファイルではそれらのフィールドが以下のフォーマットで指定されます。

```
$ExtraInfo_variable
```

variable は管理情報ベース (MIB) 変数 (ifIndex など) またはその他のデータ (NCIM テーブル内の列名など) を表します。MIB 変数は大/小文字混合で指定され、その他のデータは大文字で指定されます。例えば、次のようになります。

```
$ExtraInfo_ifIndex
$ExtraInfo_MONITOREDENTITYID
```

Tivoli Netcool/OMNIbus のプローブのルール・ファイルを構成するには、以下について理解する必要があります。

- プローブ・ルール・ファイルで使用可能な Network Manager のイベント・ソース・データ
- Network Manager のイベント・データを取り込むことのできる一連の alerts.status フィールド
- Network Manager と alerts.status のフィールドの間のデータ・マッピング

プローブ・ルール・ファイルで使用される構文については、Tivoli Netcool/OMNIbus インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.tivoli.nam.doc/welcome_ob.htm) の「*IBM Tivoli Netcool/OMNIbus* プローブとゲートウェイ・ガイド」を参照してください。

ルール・ファイル処理の例

この例は、alerts.status テーブルに挿入される出力データを生成するために、Network Manager からのソース・データがルール・ファイルによってどのように処理されるかを示します。

以下のサンプル・コードは、処理のために Tivoli Netcool/OMNIbus のプローブに渡される Network Manager のイベント・データ・レコードを示しています。このレコードでは、**ncp_ctrl** が **ncp_store** プロセスを開始したときに解決イベントが作成されています。

```
{
 EventName='ItnmServiceState';
 Severity=1;
 EntityName='BACKUP';
 Description='ncp_store process [15299] has started';
 ExtraInfo={
  EVENTTYPE=2;
  SOURCE='ncp_ctrl';
  ALERTGROUP='ITNM Status';
  EVENTMAP='ItnmStatus';
  SERVICE='ncp_store';
  PID=15299;
  };
}
```

以下に示すプローブ・ルール・ファイルの一部分は、これらの入力フィールドを処理して alerts.status のフィールドにマップするための構文を示しています。

```
...
    #
    # いくつかの標準フィールドにデータを取り込みます。
    #
    @Severity = $Severity
    @Summary = $Description
    @EventId = $EventName
    @Type = $ExtraInfo_EVENTTYPE
    @AlertGroup = $ExtraInfo_ALERTGROUP
    @NmosEventMap = $ExtraInfo_EVENTMAP
    @Agent = $ExtraInfo_SOURCE

    if (exists($ExtraInfo_ACCESSIPADDRESS))
    {
        @Node = $ExtraInfo_ACCESSIPADDRESS
```

```
    }
    else
    {
        @Node = $EntityName
    }

    #
    # イベントにその発生元ドメインの名前をスタンプとして付加します。
    #
    @NmosDomainName = $Domain
    @Manager = "ITNM"
    @Class = 8000

    #
    # RCA のフィールドにデータを取り込みます。
    #
    @LocalNodeAlias = @Node

...

    #
    # AlertKey および Identifier を設定します。
    #
    if (match(@AlertGroup, "ITNM Status"))
    {
        switch ($EventName)
        {
            case ...
...
            case "ItnmServiceState":
                @LocalPriObj = $ExtraInfo_SERVICE
...
            case ...
....
        }
    }

    #
    # Identifier と AlertKey はどちらもドメイン名を含んでいます。これにより
    # マルチドメイン・セットアップでは、イベントがドメイン単位で処理されるようになります。
    #

    #
    # AlertKey に LocalPriObj を組み込みます。そうしないと、すべてのインターフェース
    # 上のリンクダウンが、いずれかのインターフェース上のリンクアップによってクリアされます。
    #
    @AlertKey = $EntityName + @LocalPriObj + "->" + $EventName + @NmosDomainName

    #
    # 非重複化識別子を設定し、LocalPriObj を組み込みます。
    # これにより、インターフェース上で生成されたイベントの非重複化を正しく処理できるようになります。
    #
    @Identifier = $EntityName + @LocalPriObj + "->" + $EventName + @Type + @NmosDomainName
}
```

ルール・ファイルの処理が完了すると、ObjectServer に転送される出力データは以下のような形式となります。

```
CMonitorProbeApp::ProcessStatusEvent
{
 AlertGroup='ITNM Status';
 EventId='ItnmServiceState';
 Type=2;
 Severity=1;
 Summary='ncp_store process [15299] has started';
 Node='BACKUP';
 NmosDomainName='PRIMARY';
 LocalNodeAlias='BACKUP';
 LocalPriObj='ncp_store';
 LocalRootObj='';
 RemoteNodeAlias='';
 AlertKey='BACKUPncp_store->ItnmServiceStateVIRTUAL';
 Identifier='BACKUPncp_store->ItnmServiceState2VIRTUAL';
```

```
 Class=8000;
 Agent='ncp_ctrl';
 LastOccurrence=1267122089;
}
```

この例のルール・ファイル処理に基づいて、Network Manager の入力フィールドが以下のように alerts.status のフィールドにマップされます。

Network Manager フィールド	**alerts.status** フィールド
EventName	EventId
Severity	Severity
EntityName	Node
説明	Summary
ExtraInfo->EVENTTYPE	タイプ
ExtraInfo->SOURCE	エージェント
ExtraInfo->ALERTGROUP	AlertGroup
ExtraInfo->EVENTMAP	NmosEventMap
ExtraInfo->SERVICE	LocalPriObj

Network Manager イベント・データ・フィールド

Network Manager でイベントが生成されると、イベント・データが Network Manager のテーブル内の複数のフィールド (つまり列) に挿入されます。各イベントが使用するのは一部のフィールドだけですが、すべてのイベント・タイプに共通のフィールドが多数あります。

以下の表に、プローブ・ルール・ファイルで使用できるすべての Network Manager フィールド名をリストし、各フィールドに保管されるイベント・データについて説明します。またこの表では、すべてのイベントに共通であるために、ルール・ファイルで常に使用可能な Network Manager のフィールドも示します。

表 *62.* イベントのデータが取り込まれる *Network Manager* のフィールド

Network Manager フィールド名	フィールドの内容	常に使用可能か
説明	イベントの要旨	はい
Domain	現行ドメイン。 Network Manager がフェイルオーバー・モード用に構成されている場合は、これがプライマリー・ドメインとなります。	はい (マップ・ファイルが変更されていない場合)
EntityName	ネットワーク・イベントの場合は、イベントが生成されるエンティティーの NCIM entityData テーブルの entityName フィールド。 状況イベントの場合は、常にイベントが生成されるドメインの名前です。	はい
EventName	イベント ID。例えば、ItnmDiscoPhase。	はい

表 62. イベントのデータが取り込まれる *Network Manager* のフィールド (続き)

Network Manager フィールド名	フィールドの内容	常に使用可能か
ExtraInfo_ACCESSIPADDRESS	EntityName 入力フィールドで識別されるメイン・ノードまたはインターフェース・エンティティーに直接アクセス可能な IP アドレスがある場合は (NCIM インターフェース・テーブルまたはシャーシ・テーブルの accessIPAddress フィールド)、それがこのフィールドに入ります。ネットワーク・イベントにのみ適用されます。	いいえ
ExtraInfo_AGENT	ディスカバリー・エージェント (ItnmDiscoAgentStatus) イベントを担当するエージェント。	はい (ItnmDiscoAgentStatus イベントの場合)
ExtraInfo_ALERTGROUP	イベントのアラート・グループ。Network Manager 状況イベントの場合はアラート・グループが `ITNM Status` となり、ネットワーク・イベントの場合は、値が `ITNM Monitor` となります。	はい
ExtraInfo_ENTITYCLASS	NCIM の entityClass テーブルおよび classMembers テーブルに示されている、エンティティーに割り当てられたクラスの名前。	はい (ネットワーク・イベントおよび ItnmEntityCreation イベントの場合)
ExtraInfo_ENTITYTYPE	NCIM entityType テーブルに定義されている、エンティティーのタイプ。	はい (ネットワーク・イベントの場合)
ExtraInfo_LocalPriObj	alerts.status レコード内の LocalPriObj フィールドの値を指定します。このフィールドは、推奨されない ExtraInfo_EventSnmpIndex フィールドと同じ値を持ちます。ただし、ポーリング対象の MIB エンティティーの ID が接頭部として追加されます (例えば、ifEntry、bgpPeerEntry)。	はい (ネットワーク・イベントの場合)
ExtraInfo_EVENTTYPE	Network Manager によって生成されるイベントのタイプ。値は以下のとおりです。 • 1: 問題 • 2: 解決 • 13: 情報	はい
ExtraInfo_FINDER	ディスカバリー・ファインダー (ItnmDiscoFinderStatus) イベントを担当するファインダー。	はい (ItnmDiscoFinderStatus イベントの場合)
ExtraInfo_ifIndex	NCIM インターフェース・テーブルに ifIndex 値があるインターフェースに対して生成されたイベントの場合は、その値がこのフィールドに入ります。インターフェースに対するネットワーク・イベントにのみ適用されます。	いいえ

表 62. イベントのデータが取り込まれる *Network Manager* のフィールド (続き)

Network Manager フィールド名	フィールドの内容	常に使用可能か
ExtraInfo_IFALIAS	インターフェースに対して生成されたイベントの場合は、このフィールドに ifAlias 値 (認識されている場合) が入ります。ネットワーク・インターフェース・ポーリングにのみ適用されます。	いいえ
ExtraInfo_IFDESCR	インターフェースに対して生成されたイベントの場合は、このフィールドに ifDescr 値 (認識されている場合) が入ります。ネットワーク・インターフェース・ポーリングにのみ適用されます。	いいえ
ExtraInfo_IFNAME	インターフェースに対して生成されたイベントの場合は、このフィールドに ifName 値 (認識されている場合) が入ります。ネットワーク・インターフェース・ポーリングにのみ適用されます。	いいえ
ExtraInfo_IFTYPESTRING	インターフェースに対して生成されたイベントの場合は、このフィールドに ifType 値のストリング表現が入ります。ネットワーク・インターフェース・ポーリングにのみ適用されます。	いいえ
ExtraInfo_MAINNODEADDRESS	NCIM シャーシ・テーブルの accessIPAddress フィールドで識別される、エンティティーを含んだメイン・ノードの管理インターフェース。ネットワーク・イベントおよび ItnmEntityCreation イベントにのみ適用されます。	はい (ネットワーク・イベントの場合)
ExtraInfo_MAINNODEENTITYID	NCIM シャーシ・テーブルの accessIPAddress フィールドで識別される、メイン・ノードの NCIM entityData テーブルの entityId フィールド。ネットワーク・イベントにのみ適用されます。	はい (ネットワーク・イベントの場合)
ExtraInfo_MAINNODEENTITYNAME	NCIM で識別される、メイン・ノードの NCIM entityData テーブルの entityName フィールド。ネットワーク・イベントにのみ適用されます。	はい (ネットワーク・イベントの場合)
ExtraInfo_MONITOREDENTITYID	イベントが生成されるエンティティーの NCIM entityData テーブルの entityId フィールド。ネットワーク・イベントおよび ItnmEntityCreation イベントにのみ適用されます。	いいえ
ExtraInfo_MONITOREDINSTID	ncpolldata.monitoredInstance テーブル内のレコード。	いいえ
ExtraInfo_NEWPHASE	開始されたディスカバリー・フェーズ。ディスカバリー・フェーズ (ItnmDiscoPhase) イベントにのみ適用されます。	はい (ディスカバリー・フェーズ・イベントの場合)

表 62. イベントのデータが取り込まれる *Network Manager* のフィールド (続き)

Network Manager フィールド名	フィールドの内容	常に使用可能か
ExtraInfo_OLDPHASE	完了したディスカバリー・フェーズ。ディスカバリー・フェーズ (ItnmDiscoPhase) イベントにのみ適用されます。	はい (ディスカバリー・フェーズ・イベントの場合)
ExtraInfo_POLICYNAME	イベントを生成したポーリング・ポリシーの名前。	はい (ネットワーク・イベントの場合)
ExtraInfo_PID	影響を受ける Network Manager サービスのプロセス ID。 ItnmServiceState イベントにのみ適用されます。	はい (サービス状態イベントの場合)
ExtraInfo_REMOTEDOMAIN	リモート・ドメインの名前。 ItnmFailoverConnection イベントにのみ適用されます。	はい (フェイルオーバー接続イベントの場合)
ExtraInfo_sysContact	ItnmEntityCreation イベントの場合にのみ、sysContact 値が入ります (値を得られた場合)。	いいえ
ExtraInfo_sysLocation	ItnmEntityCreation イベントの場合にのみ、sysLocation 値が入ります (値を得られた場合)。	いいえ
ExtraInfo_sysObjectId	ItnmEntityCreation イベントの場合にのみ、sysObjectId 値が入ります (値を得られた場合)。	いいえ
ExtraInfo_SERVICE	影響を受ける Network Manager サービスの名前。 ItnmServiceState イベントにのみ適用されます。	はい (サービス状態イベントの場合)
ExtraInfo_SNMPSTATUS	数値の SNMP 状況コード。	はい (NmosSnmpPollFail イベントの場合)
ExtraInfo_SNMPSTATUSSTRING	人間が読める形式で示された SNMP 障害状態。	はい (NmosSnmpPollFail イベントの場合)
ExtraInfo_SOURCE	イベントの発生元であるプロセスの名前。	はい
ExtraInfo_STITCHER	ディスカバリー・スティッチャー (ItnmDiscoStitcherStatus) イベントを担当するスティッチャー。	はい (ItnmDiscoStitcherStatus イベントの場合)
Severity	イベントの重大度レベル。重大度はゼロ以外の値です。	はい

Network Manager で使用する alerts.status フィールド

ObjectServer の alerts.status テーブルには、プローブによって検出された問題の状況情報が含まれています。

標準の alerts.status フィールドのサブセットは、Network Manager イベント・データと一緒に取り込まれます。さらに、一連の専用 alerts.status フィールドは、Network Manager に固有のデータを保持するために予約されています。専用 alerts.status フィールド名は、接頭部の `Nmos` で識別できます。

以下の表では、Network Manager フィールドで取り込まれる alerts.status フィールドについて説明します。これらの alerts.status フィールドのいくつかは、プローブの規則ファイル内からの、割り振られたデフォルト値です (これらのデフォルト値は変更しないようにしてください)。

表 63. *Network Manager* で使用する *alerts.status* フィールド

alerts.status フィールド	データ型	説明	**Network Manager** フィールド名/規則ファイルでのデフォルト値
エージェント	varchar(64)	イベントを生成したプロセスの名前。このフィールドを使用すると、AEL をフィルタリングして、特定のタイプのイベント、例えば、(ncp_disco という値を持つ) ディスカバリー・イベントのみを表示できます。	ExtraInfo_SOURCE
AlertGroup	varchar(255)	イベントをタイプ別にグループ化するときに使用します。Network Manager イベントを基にデフォルトで入力されている値は、ネットワーク・イベントの場合は「ITNM Monitor」であり、状況イベントの場合は「ITNM Status」です。	ExtraInfo_ALERTGROUP
AlertKey	varchar(255)	イベントに関連するいくつかのエレメントを連結しているテキスト・ストリング。エレメントに含むことができるのは、イベント ID、ドメイン、フェーズ、およびプロセス名です。問題イベントと解決イベントを突き合わせることができます。	この値は、ObjectServer 内で問題イベントと解決イベントを適切に突き合わせることができるように、入力を基に生成されます。
Class	integer	Tivoli Netcool/OMNIbus のプローブに割り当てられているアラート・クラス。	Network Manager によって生成されたイベントに対しては、8000 という値が予約されます。
EventId	varchar(255)	イベントのタイプ (SNMPTRAP-linkDown など)。イベント・ゲートウェイは、この値を使用してイベント・マップを検索し、イベントの優先順位を決定します。	EventName
ExpireTime	integer	データベース内のイベントの有効期限時刻。現在 Network Manager では使用されていません。	
FirstOccurrence	time	イベントが最初に発生した時刻を示すタイム・スタンプ。	
ID	varchar(255)	ある特定のエンティティーでのイベントのタイプごとの固有値 (例えば、特定のデバイス・インターフェースでの LinkDown イベント)。この Identifier は、非重複化を制御します。	この値は、ObjectServer 内でイベントを適切に非重複化できるように、入力を基に生成されます。規則ファイルでは、Identifier はフィールド値の連結として構成されます。
LastOccurrence	time	イベントが最後に発生した時刻を示すタイム・スタンプ。	

表 63. *Network Manager* で使用する *alerts.status* フィールド (続き)

alerts.status フィールド	データ型	説明	**Network Manager** フィールド名/規則ファイルでのデフォルト値
LocalNodeAlias	varchar(64)	デバイスの IP または DNS アドレス。この値は、通常はシャーシのことを指しますが、pingFails の場合にのみ、インターフェースに対応することがあります。	ネットワーク・イベントの場合、このフィールドは Node フィールドと同じ値に設定されます。 状況イベントがイベント・ゲートウェイを介して Network Manager にフィード・バックされないようにするため、状況イベントの場合は値が設定されません。
LocalPriObj	varchar(255)	イベントが生成される特定のエンティティー。例えば、ifIndex、ifDescr、または ifPhysAddress などのフィールド値です。	ExtraInfo_AGENT、ExtraInfo_FINDER、ExtraInfo_ifIndex、ExtraInfo_NEWPHASE、ExtraInfo_SERVICE、または ExtraInfo_STITCHER。 ExtraInfo_ifIndex の値は、構文 `ifEntry.<ifIndex>` で示されます (例: `ifEntry.12`)。
LocalRootObj	varchar(255)	LocalPriObj フィールドで参照されるエンティティーのコンテナー。これはシャーシである必要はありませんが、例えば、シャーシ内部のスロットである場合があります。シャーシは、この場合も LocalNodeAlias を使用すれば識別できます。	
LocalSecObj	varchar(255)	イベントにより参照される 2 次オブジェクト。	ExtraInfo_OLDPHASE
Manager	varchar(64)	イベントを転送したシステムを識別する記述名。	Network Manager V3.8 以降によって生成されたイベントの場合は、`ITNM` という値が使用されます。 それより前のバージョンでは、`Omnibus` という値が使用されます。
NmosCauseType	integer	イベント状態。NMOS ゲートウェイによって取り込まれます。以下の値が可能です。 • 0: 不明 • 1: 根本原因 • 2: 副次	

表 63. *Network Manager* で使用する *alerts.status* フィールド (続き)

alerts.status フィールド	データ型	説明	**Network Manager** フィールド名/規則ファイルでのデフォルト値
NmosDomainName	varchar(64)	イベントを生成した Network Manager ネットワーク・ドメインの名前。 プライマリー・ドメインの名前はフェイルオーバー・モードで使用されます。 デフォルトでは、このフィールドは、Network Manager によって生成されるイベントの場合にのみ取り込まれます。他のプローブのソースなど、他のイベント・ソースでこのフィールドを取り込むためには、このプローブの規則ファイルを変更する必要があります。 このフィールドの値は、イベントがドメイン内にあるエンティティーに一致する場合、イベント・ゲートウェイによって取り込まれます。	Domain
NmosEntityId	integer	イベントが関連付けられるトポロジー・エンティティーを識別する固有のオブジェクト ID。このフィールドは NmosObjInst フィールドに似ていますが、ここにはさらに詳しい情報が含まれています。例えば、このフィールドにはデバイス内のインターフェースの ID を入れることができます。 ポーリング・エンジンによって生成されたイベントの場合、NmosEntityId フィールドの値は、プローブ・ルール・ファイルに取り込まれます。それ以外のイベントの場合、このフィールドの値は、エンティティーが特定されるとゲートウェイによって取り込まれます。	ExtraInfo_MONITOREDENTITYID

表 63. *Network Manager* で使用する *alerts.status* フィールド (続き)

alerts.status フィールド	データ型	説明	**Network Manager** フィールド名/規則ファイルでのデフォルト値
NmosEventMap	varchar(64)	イベントのイベント・マップ名およびオプションの優先順位 (例: PrecisionMonitorEvent.910)。優先順位は、Network Manager によるイベントの処理方法を示します。オプションの優先順位番号は、値の末尾にピリオド (.) を付け、その後に連結できます。優先順位を指定しないと、優先順位は 0 に設定されます。 **注:** この値は、同じ値を示すイベント・ゲートウェイ config.precedence テーブルに値を明示的に挿入すれば変更できます。	
NmosManagedStatus	integer	イベントが発生したネットワーク・エンティティーの管理状況。ネットワーク・エンティティーが管理対象外である場合、Network Manager のポーリングは中断され、他のソースからのイベントには「非管理 (unmanaged)」のタグが付けられます。このフィールドにより、非管理エンティティーのイベントを抽出することができます。このフィールドには以下の値が可能です。 • 0: 管理 • 1: オペレーター非管理 • 2: システム非管理 • 3: 有効範囲外	
NmosObjInst	integer	イベントが関連付けられる収容トポロジー・シャーシ・エンティティーを識別する固有のオブジェクト ID。NMOS ゲートウェイによって取り込まれます。 **ヒント:** このフィールドを使用すると、イベントの強化のためにイベントが渡されているかどうかを検出できます。	
NmosSerial	integer	現在のイベントを抑止しているイベントのシリアル番号。NMOS ゲートウェイによって取り込まれます。	

表 63. *Network Manager* で使用する *alerts.status* フィールド (続き)

alerts.status フィールド	データ型	説明	**Network Manager** フィールド名/規則ファイルでのデフォルト値
Node	varchar(64)	イベントの発生元であるデバイス。アクセス可能な IP アドレスを持つエンティティーに対してイベントが発生した場合は、IP アドレスが使用されます。それ以外の場合は、NCIM からの entityName 値が使用されます。デフォルトでは、Node に LocalNodeAlias と同じ値があります。	ExtraInfo_ACCESSIPADDRESS または EntityName EntityName 値が Node フィールドにマップされるのは、ExtraInfo_ACCESSIPADDRESS 入力フィールドが空の場合に限ります。
NodeAlias	varchar(64)	メイン・ノードの IP アドレス (使用可能な場合)。	ExtraInfo_MAINNODEADDRESS
RemoteNodeAlias	varchar(64)	リモート・ノードのネットワーク・アドレス (関係があるもの)。例: • 空の値 (インターフェースがダウンした場合) • 隣接するアドレス (接続されているインターフェースがダウンした場合) • ポーリング・ステーション (ping 障害イベントの場合)	
Serial	incr	各 ObjectServer インスタンスのイベントごとの固有 ID。 プライマリー ObjectServer およびバックアップ ObjectServer がどこで構成されるかにより、ObjectServer には、同じイベントに対して異なるシリアル番号が付きます。	
ServerName	varchar(64)	送信側の ObjectServer の名前。	
ServerSerial	integer	送信側の ObjectServer でのイベントのシリアル番号。 プライマリー ObjectServer およびバックアップ ObjectServer がどこで構成されるかにより、ObjectServer には、同じイベントに対して異なるシリアル番号が付きます。イベントが現在の ObjectServer で発生した場合、ServerSerial の値は、Serial の値と同じになります。	

表 63. *Network Manager* で使用する *alerts.status* フィールド (続き)

alerts.status フィールド	データ型	説明	**Network Manager** フィールド名/規則ファイルでのデフォルト値
Severity	integer	ObjectServer に保管されているイベントの重大度レベル。以下の値がデフォルトです。 • 0: クリア (緑) • 1: 不確定 (紫) • 2: 警告 (青) • 3: 軽微 (黄) • 4: 重要 (オレンジ) • 5: 重大 (赤)	Severity
StateChange	time	イベントが最後に変更された日時を示すタイム・スタンプ。このフィールドは、あるプロセスが ObjectServer に追加された後に、そのプロセスがイベントを変更するかどうかを調べるために使用できます。	
Summary	varchar(255)	イベントのテキスト形式の説明。	説明
Tally	integer	イベントが発生した回数。この値は、イベント・リストまたは AEL の「カウント」列と、mojo.events テーブルの「発生済み」列に表示されます。	
タイプ	integer	アラートのタイプ。Network Manager に対する特定の関連性を表す値は、以下のとおりです。 • 1: 問題 • 2: 解決 • 13: 情報	ExtraInfo_EVENTTYPE

alerts.status テーブルについて詳しくは、Tivoli Netcool/OMNIbus インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.tivoli.nam.doc/welcome_ob.htm) の『*IBM Tivoli Netcool/OMNIbus* 管理ガイド』を参照してください。

付録 F. Network Manager のイベント・カテゴリー

Network Manager によって生成されるイベントは、2 つのカテゴリーに分類されます。1 つはモニターされているネットワークに関するイベント、もう 1 つは Network Manager のプロセスに関するイベントです。

これらのイベントは、Tivoli Netcool/OMNIbus ObjectServer に保管されます。Tivoli Netcool/OMNIbus のプローブ (**nco_p_ncpmonitor**) を使用して、イベント・データが処理され、ObjectServer 内の alerts.status テーブルに転送されます。

以下の図は、Network Manager から ObjectServer へのイベントのフローを示しています。

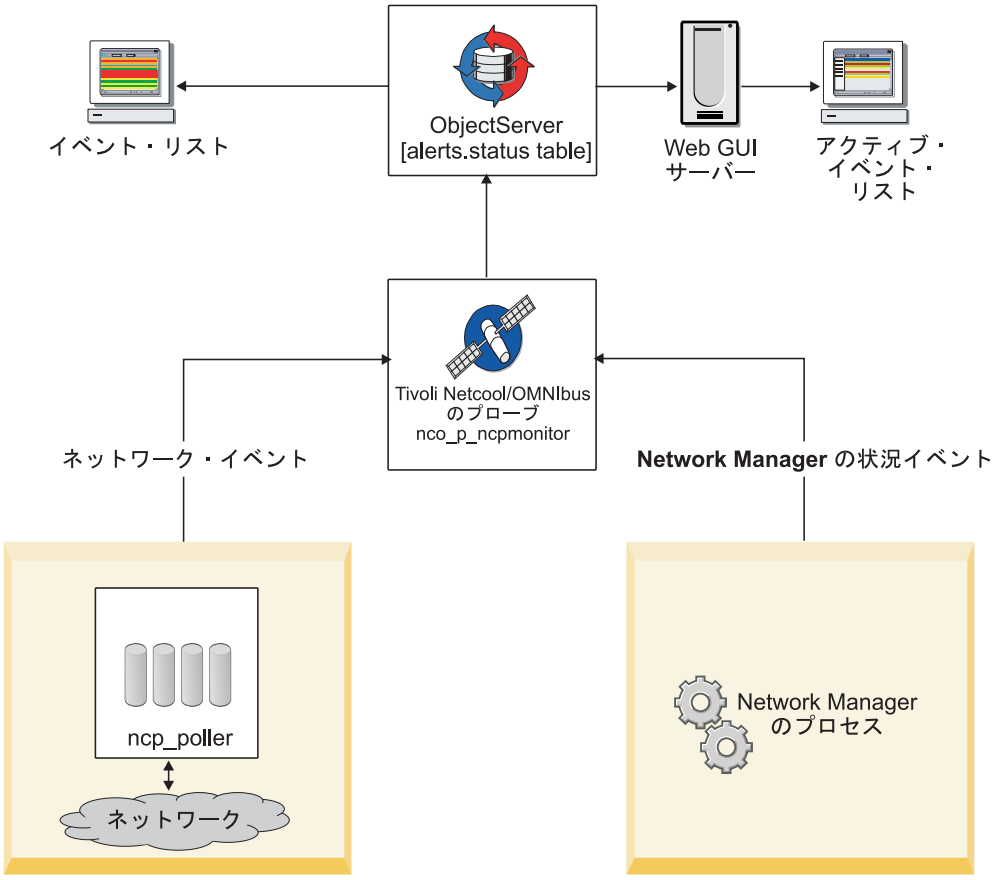


図 *15. Network Manager から Tivoli Netcool/OMNIbus へのイベントのフロー*

関連資料:

116 ページの『デフォルトのイベント・マップ』
Network Manager には、デフォルトのイベント・マップ・セットが用意されています。ここでは、使用可能なデフォルトのイベント・マップおよび各イベント・マップの仕組みについて説明し、さらに 3.9 イベント・マップがレガシー・イベント・マップを代行する方法を説明します。

© Copyright IBM Corp. 2006, 2016

Network Manager ネットワーク・イベント

ポーリング・エンジン **ncp_poller** は、ネットワークの状態に関するイベントを生成します。これらのイベントは、ネットワークの問題を識別するために使用でき、ネットワーク・ポーリング GUI (「管理」 > 「ネットワーク」 > 「ネットワーク・ポーリング」の順に移動) を使用して構成できます。これらのイベントはネットワーク・イベントと呼ばれ、alerts.status の AlertGroup フィールドで ITNM Monitor という値を割り振られます。

各ネットワーク・イベントは、インターフェースやシャーシなどの単一のエンティティーで生成され、イベント・データはポーリングのタイプによって異なります。ネットワーク・イベントが ObjectServer に転送され、alerts.status テーブルに挿入されると、 AlertGroup 値として「ITNM Monitor」が割り振られます。

ネットワーク・イベントには、イベント ID を無限に使用できます。SNMP ポーリングが失敗したときに生成されるイベントには、alerts.status テーブルで EventID 値として NmosSnmpPollFail が割り振られます。

ObjectServer 内のネットワーク・イベントは、イベント・ゲートウェイを介して Network Manager に戻されて、根本原因分析を含むイベント・エンリッチが実行されます。

Network Manager 状況イベント

Network Manager は、Network Manager の各種のプロセスの状況を示すイベントを生成できます。このイベントは Network Manager 状況イベントと呼ばれ、alerts.status の AlertGroup フィールドで ITNM Status という値を割り振られます。

これらの状況イベントが ObjectServer に転送され、alerts.status 表に挿入されると、 AlertGroup 値として「ITNM Status」が割り振られます。

状況イベント・タイプ

Network Manager 状況イベントをタイプ別に示すために、一連のイベント ID が使用されます。 alerts.status 表に挿入される EventId 値と、それぞれに関連する状況イベントがどのように生成されるのかを以下のリストに示します。

ItnmDatabaseConnection

このタイプのイベントは、NCIM への接続が失われたことを示すために生成されます。このイベントは、**ncp_model** プロセスで管理状況ポーリング・スレッドによって生成されます。このイベントの生成は、モデル内の管理状況ポーリング間隔で構成されている時間に依存します。接続が失われると問題イベントが生成され、接続が復元された場合または始動時に、対応する解決イベントが生成されて、前の操作での障害がすべて消去されます。このイベント・タイプは、フェイルオーバーが構成されている場合に、バックアップ・ドメインによる引き継ぎを可能にします。仮想ドメイン・プロセスは、NCHOME/etc/precision/VirtualDomainSchema.cfg ファイル内の NCIM に対するフィルターで定義されているとおりに、このイベントに対応します。

ItnmDiscoAgentStatus

このタイプのイベントは、ディスカバリー・エージェントが新しい状態に移行したときに **ncp_disco** によって生成されます。ディスカバリーの終了時、ディスカバリー中に使用されたエージェントごとに情報イベントが ObjectServer に転送されます。

この情報を使用すると、各エージェントの状態を識別できます。alerts.status 表では、LocalPriObj フィールドを使用してエージェントの名前が保管されます。

ObjectServer のディスカバリー・エージェント・イベントは、後続のディスカバリーが実行されると上書きされます。

ItnmDiscoFinderStatus

このタイプのイベントは、ディスカバリー・ファインダーが新しい状態に移行したときに **ncp_disco** によって生成されます。ディスカバリーの終了時、ディスカバリー中に使用されたファインダーごとに情報イベントが ObjectServer に転送されます。

この情報を使用すると、実行中のファインダーとその状態を識別できます。alerts.status 表では、LocalPriObj フィールドを使用してファインダーの名前が保管されます。

ObjectServer のディスカバリー・ファインダー・イベントは、後続のディスカバリーが実行されると上書きされます。

ItnmDiscoPhase

このタイプのイベントは、ディスカバリー・プロセスが新しいフェーズに移行したときに **ncp_disco** によって生成されます。ディスカバリーの終了時、ループ移行 (フェーズ 0 (スタンバイ) →フェーズ 1、2、3 (データ収集) →フェーズ -1 (データ処理)) を示す 5 つの情報イベントが ObjectServer に存在する必要があります。単一のディスカバリーでは、以下のようにフェーズが変化するたびにイベントが生成されます。

- 0 から 1
- 1 から 2
- 2 から 3
- 3 から -1
- -1 から 0

この情報を使用すると、各フェーズの長さを判別できます。 alerts.status 表では、LocalPriObj フィールドを使用してディスカバリーの移行先のフェーズが保管され、 LocalSecObj フィールドにはディスカバリーの前のフェーズが保管されます。

ヒント: **ncp_disco** プロセスをデバッグ・モードで実行したときは、ディスカバリー・ログ・ファイルにフェーズのストリング値も表示されます。

ObjectServer のディスカバリー・フェーズ・イベントは、後続のディスカバリーが実行されると上書きされます。

ItnmDiscoStitcherStatus

ディスカバリー・プロセスは、データ収集ステージと、データ処理ステージから構成されます。データ処理ステージでは、トポロジーが作成されます。

データ処理ステージで主要なフェーズに達すると、ディスカバリー・エンジン **ncp_disco** によって、ItnmDiscoStitcherStatus イベントが生成されます。ディスカバリーの終了時、ディスカバリー中に使用された重要なディスカバリー・スティッチャーごとに情報イベントが ObjectServer に転送されます。

この情報を使用して、ディスカバリーがデータ処理ステージのどのフェーズにあるかを識別できます。 alerts.status 表では、LocalPriObj フィールドを使用して、このフェーズに対応するスティッチャーの名前が保管されます。

以下のスティッチャーが実行を開始すると、ItnmDiscoStitcherStatus イベントが発生します。

- BuildFinalEntityTable
- BuildContainment
- BuildLayers
- MergeLayers
- PostLayerProcessing

以降のイベントは、トポロジー作成フェーズの間に、以下のスティッチャーが実行されると発生します。

- CreateScratchTopology
- PostScratchProcessing
- SendTopologyToModel

ObjectServer のディスカバリー・スティッチャー・イベントは、後続のディスカバリーが実行されると上書きされます。

ItnmEntityCreation

$NCHOME/etc/precision/ModelSchema.cfg ファイルで構成されている場合、このタイプの情報イベントは、新しいシャーシまたは IP インターフェース・エンティティー (EntityType = 1) が NCIM データベースに挿入されるたびに **ncp_model** によって生成されます。

ModelSchema.cfg は、model.config テーブルの INSERT ステートメントで RaiseEntityEvent 列の値に 1 を設定することで構成できます。例:

```
create table model.config
(
LingerTime int not null primary key,          // デフォルト値 3 (ディスカバリー)
RaiseEntityEvent int type boolean not null,   // デフォルト値 0 (オフ)
DiscoveryUpdateMode int not null,             // デフォルト値 0 - フル・ディスカバリー、
                                              //              1 - 部分的なディスカバリー
unique(LingerTime)
);
insert into model.config values (3, 1, 0);
```

注: 構成の変更を反映して、イベントを有効にするには、**ncp_model** プロセスを再始動する必要があります。このプロセスは、始動時に構成設定を読み取ります。

ItnmEntityDeletion

$NCHOME/etc/precision/ModelSchema.cfg ファイルで構成されている場合、このタイプの情報イベントは、シャーシまたは IP インターフェース・エンティティー (EntityType = 1) が NCIM データベースから削除されるたびに **ncp_model** によって生成されます。

ModelSchema.cfg は、model.config テーブルの INSERT ステートメントで RaiseEntityEvent 列の値に 1 を設定することで構成できます (前の ItnmEntityCreation EventId の説明を参照)。

ItnmFailover

このタイプのイベントは、フェイルオーバー・ペア内の Network Manager ドメインがフェイルオーバーまたはフェイルバックしたときに **ncp_virtualdomain** によって生成されます。

フェイルオーバーの発生時に問題イベントが生成され、フェイルバック時に解決イベントが生成されます。

alerts.status テーブルの「要約」フィールドの説明は、ドメインが 1 次とバックアップのいずれであるのか、およびドメインがアクティブ・モードとスタンバイ・モードのいずれであるのかを示しています。

ItnmFailoverConnection

このタイプのイベントは、フェイルオーバー・ペアのバックアップ・ドメインが 1 次ドメインにいつ接続または切断したのかを示すために **ncp_virtualdomain** によって生成されます。

Network Manager がフェイルオーバー・モードで稼働しているときは、 1 次ドメインとバックアップ・ドメインが TCP ソケット接続をセットアップした時点で解決イベントが生成されます。ディスカバリー・プロセス (**ncp_disco**) がバックアップ・ドメインで稼働していないので、トポロジーの更新情報を 1 次ドメインから転送するには、このソケット接続が必要です。接続がその後失われた場合は、問題イベントが生成されます。

注**:** フェイルオーバーが起動されたかどうかは、接続の状況から判別できません。フェイルオーバーが起動されるのは、正常性検査イベントが (ObjectServer 経由で) ドメインに転送され、ソケット接続がある時点で確立された場合に限られます。

ItnmHealthChk

Network Manager のフェイルオーバーは、正常性検査イベントによって管理されます。フェイルオーバー・ペアの各ドメインは、ドメインが正常である間、正常性検査解決イベントを生成します。

ドメインの正常性検査問題イベントは、以下の 2 つの方法で生成されます。

- ローカル・ドメインによって: ローカル・ドメインは、そのいずれかのプロセスの障害を、 $NCHOME/etc/precision/VirtualDomainSchema.cfg ファイルの構成に従って検出します。
- リモート・ドメインによって: 他方のドメインが正常性検査解決イベントを所定の時間生成していないことを一方のドメインが検出すると、一方のドメインはリモート・ドメインに代わって合成正常性検査問題イベントを生成します。

プライマリー・ドメインで正常性検査問題イベントが生成されると、フェイルオーバーが開始され、バックアップ・ドメインがアクティブになります。

以前、正常性検査イベントには、EventID 値として NcpHealthChk が割り振られていました。以前のバージョンの Network Manager との互換性を

確保するには、プローブ・ルール・ファイルの ItnmHealthChk を NcpHealthChk で置き換えてください。

注: 正常性検査イベントは、Network Manager イベント・ゲートウェイによって処理されます。このイベント・ゲートウェイでは、イベントが参照するドメインをノード値に設定する必要があります。これは、イベントを生成するドメインにする必要はありません。なぜなら、一方のドメインが他方のドメインに代わって障害イベントを生成できるからです。

ItnmMaintenanceState

$NCHOME/etc/precision/ModelSchema.cfg ファイルで構成されている場合、このタイプのイベントは、シャーシまたは IP インターフェースの保守状況が変化したときに、トポロジー・マネージャー、 **ncp_model** によって生成されます。

ModelSchema.cfg は、model.config 表の INSERT ステートメントで RaiseEntityEvent 列の値に 1 を設定することで構成できます (前の ItnmEntityCreation イベントの説明を参照)。

シャーシまたは IP インターフェース・エンティティーが保守になると問題イベントが生成され、エンティティーが保守でなくなると解決イベントが生成されます。

注: インターフェース・イベントが個々に送信されるのは、変更がシャーシ・レベルで適用されない場合に限られます。デバイスが変更されると、シャーシ・イベントと一連のインターフェース・イベントはまとめて生成されません。

ItnmServiceState

このタイプのイベントは、プロセスが開始または終了したときに生成され、プロセスが開始できなかったのか、それとも実行時にプロセスが停止したのかを示します。 (システム・シャットダウン時にプロセスが停止したときは、プロセス状態イベントが生成されないことに注意してください。)

ncp_ctrl がプロセスを開始したときは、解決イベントが生成されます。プロセスが開始できなかった場合、または実行時にプロセスが停止した場合は、問題イベントが生成されます。

alerts.status 表の「要約」フィールドの説明には、プロセス名と PID が含まれており、プロセスが以下の状態であるかどうかが示されています。

- 開始済み (正常に初期化されています)
- 停止 (services.inTray という名前の **ncp_ctrl** データベース表から削除されています)
- 終了済み (停止したが、**ncp_ctrl** によって再始動されます)
- 始動に失敗
- 失敗したが、再始動はしない (停止後に、プロセスに構成した再試行回数を超過しました)

ItnmTopologyUpdated

このタイプの情報イベントは、NCIM トポロジー・データベースの更新がディスカバリー・サイクルの最後で完了したときに **ncp_model** によって生

成されます。この情報は、NCIM データベースの更新後に実行するスクリプトまたはプロシージャーをプログラムする場合に役立ちます。

注: フィードバック・オプションをオンにするか、または大きなサブネットを ping する場合は、ディスカバリー・サイクルとこのタイプのイベントが複数存在する場合があります (ディスカバリー・サイクルごとに 1 つのイベントが存在します)。ディスカバリーが最終的に完了したかどうかを判別するには、 Ping ファインダー・サービスに対して以下の OQL 照会を実行します。

```
select * from pingFinder.status where m_Completed <> 1;
```

この照会では、Ping ファインダーがまだ ping しているサブネットがないかどうかが検索されます。未解決の ping スイープがなく、ディスカバリーのフェーズが 0 の場合は、ディスカバリーが完了しています。

付録 G. ポーリング・データベース

ここでは、ポーリングに使用されるデータベースの構造について説明します。

NCMONITOR データベース

NCMONITOR スキーマは、ポーリングで使用される複数のデータベースをホストします。

ncmonitor データベース内のポーリング用の SNMP テーブル

ncmonitor データベース内の SNMP テーブルは、ポーリング・エンジン (ncp_poller) によって使用され、SNMP を使用してディスカバーされた各デバイスへのアクセス方法に関する情報を保管します。

ncp_dh_snmp プロセス、および ncp_poller プロセスはいずれも ncmonitor データベースを使用します。ただし、データベースにデータを取り込むのは ncp_dh_snmp プロセスのみです。ncp_poller プロセスは、データベースを読み取り専用として扱います。このため、SNMP を使用してデバイスをディスカバーし、そのデバイスを SNMP を使用してモニターしている必要があります。

ncmonitor データベースは、`$NCHOME/etc/precision/DbLogins.DOMAIN.cfg` (ここで、*DOMAIN* はディスカバーされたデバイスが含まれているドメイン) で定義されています。

ncmonitor データベースには、以下のテーブルがあります。

- ncmonitor.snmpTarget
- ncmonitor.snmpAccess
- ncmonitor.snmpv1Sec
- ncmonitor.snmpv3Sec
- ncmonitor.snmpUser

ncmonitor.snmpTarget テーブル

snmpTarget テーブルには、Network Manager が認識している各 IP アドレスがリストされます。

表 *64. ncmonitor.snmpTarget* データベース表

列名	制約	データ型	説明
targetid	• 基本キー • NULL 以外	テキスト	ターゲットを示す固有 ID。
netaddr		テキスト	ターゲットの IP アドレス。

© Copyright IBM Corp. 2006, 2016

表 64. *ncmonitor.snmpTarget* データベース表 (続き)

列名	制約	データ型	説明
readaccessid	外部キー	テキスト	snmpaccess テーブルを参照してください。このターゲットに対して、SNMP Get 動作および GetNext 動作を実行するためのアクセスの詳細を規定します。
writeaccessid	外部キー	テキスト	snmpaccess テーブルを参照してください。このターゲットに対して、SNMP Set 動作を実行するために使用されるアクセスの詳細を規定します。
snmpgetbulk		Boolean Integer	該当する場合 (SNMPv2 または SNMPv3 を使用するときや、テーブル・ウォークを実行するとき) に、GetNext 動作を試行するかどうかを示すフラグ。
snmpthrottleid		テキスト	テーブル・ウォーク動作を実行するときに、このターゲットに対して要求が出される速度を制御するためのスロットルの詳細。
createtime		テキスト	このターゲットの作成時刻を記録するタイム・スタンプ。
lastupdate		テキスト	このターゲットの詳細の最終変更時刻を記録するタイム・スタンプ。
domain		テキスト	このターゲットが属するドメイン。

ncmonitor.snmpAccess テーブル

snmpAccess テーブルは、SNMP アクセスの詳細を規定します。

表 65. *ncmonitor.snmpAccess* データベース表

列名	制約	データ型	説明
accessid	• 基本キー • NULL 以外	テキスト	SNMP アクセスの詳細を示す固有 ID。
version		列挙値	使用される SNMP のバージョン。想定される値は以下のとおりです。 • 0: SNMPv1 • 1: SNMPv2 • 3: SNMPv3
remoteport		整数	SNMP パケットが送信される UDP ポート。
retries		整数	中止するまでの再試行回数。
timeout		整数	SNMP 要求を再試行するまでのミリ秒数。

表 65. *ncmonitor.snmpAccess* データベース表 (続き)

列名	制約	データ型	説明
accesslevel		列挙値	提供されるアクセス権限のレベルを示すフラグ。想定される値は以下のとおりです。 • 1: 読み取り • 2: 書き込み

ncmonitor.snmpv1Sec テーブル

snmpv1Sec テーブルには、snmpAccess テーブルの SNMPv1 および SNMPv2 に関連する行のデータのみが取り込まれます。

表 66. *ncmonitor.snmpv1Sec* データベース表

列名	制約	データ型	説明
accessid	外部キー	テキスト	提供された SNMPv1 または SNMPv2 固有の詳細に対する、それぞれの詳細を参照します。
community		テキスト	当該詳細を使用して要求を送信するときに使用するコミュニティー・ストリング。
encrypted		Boolean Integer	コミュニティー・ストリングが暗号化されるかどうかを示すフラグ。可能な値は次のとおりです。 • 0: 暗号化されない • 1: 暗号化される

ncmonitor.snmpv3Sec テーブル

snmpv3Sec テーブルには、snmpAccess テーブルの SNMPv3 に関連する行のデータのみが取り込まれます。

表 67. *ncmonitor.snmpv3Sec* データベース表

列名	制約	データ型	説明
accessid	外部キー	テキスト	提供された SNMPv3 固有の詳細に対する詳細を参照します。
userid	外部キー	テキスト	snmpusmuser テーブルの userid フィールドを参照します。これは、当該詳細を使用して SNMP 要求を送信するときに使用するユーザーです。
securitylevel		列挙値	SNMPv3 のセキュリティー・レベルを示すフラグ。想定される値は以下のとおりです。 • noAuthNoPriv • authNoPriv • authPriv

表 67. *ncmonitor.snmpv3Sec* データベース表 (続き)

列名	制約	データ型	説明
defaultcontext		テキスト	ディスカバリー・エージェントによって明示的に指定されていないときに使用される SNMPv3 のコンテキスト名。

ncmonitor.snmpUser テーブル

snmpUser テーブルには、SNMPv3 プロトコルで使用される SNMP ユーザーの詳細がリストされます。

表 68. *ncmonitor.snmpUser* データベース表

列名	制約	データ型	説明
userid	• 基本キー • NULL 以外	テキスト	このユーザーを示す固有 ID。
username		テキスト	USM ユーザー名。
authpass		テキスト	認証パスワード。
privpass		テキスト	プライバシー・パスワード。これは、SNMPv3 ペイロードの暗号化に使用されます。
authtype		テキスト	SNMPv3 の認証ヘッダーに使用される暗号化方式。
privtype		テキスト	ペイロードに使用される暗号化方式。
encrypted		Boolean Integer	authpass フィールドおよび authtype フィールドが暗号化されることを示すフラグ。可能な値は次のとおりです。 • 0: 暗号化されない • 1: 暗号化される

ping ポーリング状況テーブル

NCMONITOR ping ポーリング状況テーブルを使用して、ネットワーク ping ポーリングで診断操作を実行することができます。

expectedIps 表

expectedIps 表には、特定のドメインについて Network Manager でディスカバーされることが予想される IP アドレスのリストが含まれています。この表には、ncp_upload_expected_ips.pl スクリプトを使用してデータが取り込まれます。

以下の表に、expectedIps 表の各列をリストします。

表 69. *ncmonitor.expectedIps* データベース表

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
domainMgrId	NCIM トポロジー・データベースの domainMgr 表に示されている、関連するドメインの ID。

pollLog 表

pollLog 表は、ポーリング・エンジンの状況の最新のスナップショットを保管します。この表には、ncp_ping_poller_snapshot.pl スクリプトを使用してデータが取り込まれます。このスクリプトは、ncp_poller を照会し、結果をこの表に転送します。

この表の各行は、単一のアクティブなポーリング・ポリシーの定義済みスコープ内にある単一のエンティティーに対応します。表内のフィールドは、3 つの概念的なグループに分けることができます。

『エンティティー情報』
: このエンティティー情報は、他の NCIM トポロジー・データベース表との相互参照に使用できます。

254 ページの『管理状況情報』
: これは、ポーリング・ポリシーによって適用されている管理状況です。

255 ページの『最新のポーリング状態情報』
: これは、現在のエンティティーとポリシーでの最新のポーリング状態です。

エンティティー情報

エンティティー情報を保管する pollLog 表内のフィールドを以下で説明します。

表 70. *ncmonitor.pollLog* データベース表のエンティティー情報フィールド

列名	説明
entityId	NCIM トポロジー・データベースの entityData 表に定義されている、ping の対象となるエンティティーの ID。
policyId	NCMONITOR ポリシー表に定義されている、関連する ping ポリシーの ID。
mainNodeEntityId	NCIM トポロジー・データベースの entityData 表に定義されている、エンティティーが属しているメイン・ノードの ID。シャーシ ping ポーリングの場合は、entityId と同じです。インターフェース ping ポーリングの場合は、インターフェースを含んでいるメイン・ノードの ID です。
entityType	NCIM トポロジー・データベースの entityType 表で定義されているように、このフィールドは以下のいずれかの値になります。 • 1: シャーシ ping ポーリング • 2: インターフェース ping ポーリング

表 70. *ncmonitor.pollLog* データベース表のエンティティー情報フィールド (続き)

列名	説明
ip	ICMP ping パケットの送信先 IP アドレス。これは、entityId で識別されるインターフェース・エンティティーまたはシャーシ・エンティティーの accessIPAddress です。
mainNodeAddress	NCIM トポロジー・データベースの entityData 表に示されている、エンティティーが属しているメイン・ノードの IP アドレス。これは、mainNodeEntityId で識別されるシャーシ・エンティティーの accessIPAddress です。
ifIndex	インターフェース ping ポーリングで、関連するインターフェースの ifIndex を使用できる場合があります。どの ping ポーリングでも、このフィールドは NULL になる可能性があります。
domainMgrId	NCIM トポロジー・データベースの domainMgr 表に示されている、関連するドメインの ID。

管理状況情報

管理状況情報を保管する pollLog 表内の各フィールドを以下で説明します。

表 71. *ncmonitor.pollLog* データベース表の管理状況情報フィールド

列名	説明
isManaged	エンティティーがこのポリシーでポーリングされているかどうかを示します。 • 0: False • 1: True
entityStatus	ポーリング・プログラムが使用している entityId で識別されるエンティティーの管理状況。管理されている場合は、NCIM managedStatus 表に明示的にリストされているかどうかにかかわらず、このフィールドの値が 0 に設定されます。 **注:** これはスナップショットの時点での状況であるため、NCIM managedStatus 表の内容が動的に変更された場合は、その内容と異なる可能性があります。
mainNodeStatus	ポーリング・プログラムが使用している mainNodeEntityId で識別されるエンティティーの管理状況。非管理のメイン・ノード内のインターフェースは、同じく非管理です。管理されている場合は、NCIM managedStatus 表に明示的にリストされているかどうかにかかわらず、このフィールドの値が 0 に設定されます。 **注:** これはスナップショットの時点での状況であるため、NCIM managedStatus 表の内容が動的に変更された場合は、その内容と異なる可能性があります。

表 71. *ncmonitor.pollLog* データベース表の管理状況情報フィールド (続き)

列名	説明
entityChangeTime	entityStatus フィールドに対する最終変更のタイム・スタンプ。使用されない場合は、デフォルトでゼロ・タイム・スタンプに設定されます。
mainNodeChangeTime	mainNodeStatus フィールドに対する最終変更のタイム・スタンプ。使用されない場合は、デフォルトでゼロ・タイム・スタンプに設定されます。

最新のポーリング状態情報

最新のポーリング状態情報を保管する pollLog 表内の各フィールドを以下で説明します。

表 72. *ncmonitor.pollLog* データベース表の最新のポーリング状態情報フィールド

列名	説明
lastPollFailure	ping ポーリング障害イベントが最後に生成された時刻。ポーリング・プログラムが開始されてからポーリング障害が発生していない場合は、デフォルトでゼロ・タイム・スタンプに設定されます。
lastPollInterval	最後に完了したポーリング周期の所要時間 (秒単位)。ポリシーがエンティティーをアクティブにモニターしていない場合や、ポーリング・プログラムが開始されてからまだポーリング周期が完了していない場合は、このフィールドが NULL になります。 **注:** スナップショットを作成する場合は、システムが 3 回目のポーリング間隔にあることが必要です。
timeSinceLastPoll	スナップショットが作成された時点での、最後のポーリングからの秒数。
snapshotTime	ポーリング・プログラムからデータが取得された時刻。ポリシーがエンティティーをアクティブにモニターしていない場合は、ゼロ・タイム・スタンプになります。

pollLogSummary 表

この表は、ドメインの pollLog 表に書き込まれた各スナップショットの結果から後続の各セクションで説明するビューを使用して生成される要約を保管します。この表には、ncp_ping_poller_snapshot.pl スクリプトを使用してデータが取り込まれます。このスクリプトは、ポーリング・プログラムを照会し、結果をこの表に転送します。

以下の表に、pollLogSummary 表の各列をリストします。

注**:** ncp_ping_poller_snapshot.pl スクリプトは、この表の既存のデータを消去しないため、表が増大する可能性があります。不要となったデータは、必要に応じて除去できます。そのためには、domainMgrId または summaryTimestamp フィールドに対してフィルターを適用します。

表 73. *ncmonitor.pollLogSummary* データベース表

列名	説明
domainMgrId	NCIM の domainMgr 表に示されている、関連するドメインの ID。
domainName	domainMgrId によって識別されるドメインの名前。
undiscoveredIps	pollLog にスナップショットがロードされた後に、このドメインの undiscoveredIps ビューから返された IP アドレスの数。
unmonitoredIps	pollLog にスナップショットがロードされた後に、このドメインの unmonitoredIps ビューから返された IP アドレスの数。
unmanagedIps	pollLog にスナップショットがロードされた後に、このドメインの unmanagedIps ビューから返された IP アドレスの数。
unpolledFor15MinutesIps	pollLog にスナップショットがロードされた後に、このドメインの unpolledFor15MinutesIps ビューから返された IP アドレスの数。
delayedPollPolicies	pollLog 表にスナップショットがロードされた後に、このドメインの delayedPollPolicies ビューから返された IP アドレスの数。
summaryTimestamp	要約が生成された時刻を示すタイム・スタンプ。

undiscoveredIps ビュー

undiscoveredIps ビューには、ディスカバーされることが予想されたにもかかわらず、Network Manager でディスカバーされなかったために NCIM トポロジー・データベースに追加されていない IP アドレスがリストされます。このテーブルにリストされる IP アドレスは、expectedIps テーブルにはロードされていますが、NCIM には存在しません。

以下の表に、undiscoveredIps ビューの列をリストします。

表 74. *ncmonitor undiscoveredIps* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。
domainName	domainMgrId 列によって識別されるドメイン名。

unmonitoredIps ビュー

unmonitoredIps ビューは、pollLog テーブルの最新のポーリング・プログラム・スナップショットを使用して、アクティブな ping ポリシーのスコープ内にないために現在ポーリングされていない IP アドレスを、expectedIps テーブルからリストします。

以下の表に、unmonitoredIps ビューの列をリストします。

表 75. *ncmonitor unmonitoredIps* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
mainNode	NCIM entityData テーブルに示されている、メイン・ノードの entityName。
mainNodeEntityId	NCIM entityData テーブルに示されている、メイン・ノードの entityId。
entityId	NCIM entityData テーブルに示されている、インターフェースの entityId。
class	NCIM classMembers および entityClass テーブルで定義される、メイン・ノードの entityClass。
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。
domainName	domainMgrId 列によって識別されるドメイン名。

unmanagedIps ビュー

unmanagedIps ビューは、pollLog テーブルの最新のポーリング・プログラム・スナップショットを使用して、アクティブな ping ポリシーのスコープ内にあるが非管理であるためにモニターされていない IP アドレスを、expectedIps テーブルからリストします。これは、スナップショットの時点でポーリング・プログラムに認識された管理状況に基づきます。

以下の表に、unmanagedIps ビューの列をリストします。

表 76. *ncmonitor unmanagedIps* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
mainNode	NCIM entityData テーブルに示されている、メイン・ノードの entityName。
mainNodeEntityId	NCIM entityData テーブルに示されている、メイン・ノードの entityId。
entityId	NCIM entityData テーブルに示されている、インターフェースの entityId。
class	NCIM classMembers および entityClass テーブルで定義される、メイン・ノードの entityClass。

表 76. *ncmonitor unmanagedIps* ビュー (続き)

列名	説明
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。
domainName	domainMgrId 列によって識別されるドメイン名。

unpolledFor15MinutesIps ビュー

unpolledFor15MinutesIps ビューは、pollLog テーブルの最新のポーリング・プログラム・スナップショットを使用して、直前の 15 分間にまったく ping ポーリングされていない IP アドレスを、expectedIps テーブルからリストします。これには、非管理の IP アドレスや、構成済み ping ポーリング・ポリシーのスコープ内にない IP アドレスが含まれます。

以下の表に、unpolledFor15MinutesIps ビューの各列をリストします。

表 77. *ncmonitor unpolledFor15MinutesIps* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
mainNode	NCIM entityData テーブルに示されている、メイン・ノードの entityName。
mainNodeEntityId	NCIM entityData テーブルに示されている、メイン・ノードの entityId。
entityId	NCIM entityData テーブルに示されている、インターフェースの entityId。
class	NCIM classMembers および entityClass テーブルで定義される、メイン・ノードの entityClass。
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。
domainName	domainMgrId 列によって識別されるドメイン名。

delayedPollPolicies ビュー

delayedPollPolicies ビューは、pollLog テーブルの最新のポーリング・プログラム・スナップショットを使用して、予定より遅れているアクティブな ping ポリシーをすべてリストします。

ポーリング周期の現在または直前の時間間隔 (ポーリング・プログラム・スナップショットが作成された時点から測定) が、構成済みポーリング間隔の 2 倍以上である場合は、エンティティーおよび関連するポリシーがこのビューにリストされます。この場合、ポーリング・スコープの範囲外にあるエンティティー (unmonitoredIps ビューを参照) や、管理されていないエンティティー (unmanagedIps ビューを参照) は除外され、pollLog テーブル内の最新のスナップショットのみが対象となります。

以下の表に、delayedPollPolicies ビューの各列をリストします。

表 78. *ncmonitor delayedPollPolicies* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
mainNode	NCIM entityData テーブルに示されている、メイン・ノードの entityName。
mainNodeEntityId	NCIM entityData テーブルに示されている、メイン・ノードの entityId。
entityId	NCIM entityData テーブルに示されている、インターフェースの entityId。
entityType	NCIM の entityType テーブルで定義されているように、この列には以下の値が入ります。 • シャーシ ping ポーリングの場合は 1 • インターフェース ping ポーリングの場合は 2
policy	NCMONITOR ポリシー・テーブルに示されている、ping ポリシーの policyName。
configuredPollInterval	NCMONITOR ポリシー・テーブルに示されている、ping ポリシーの構成済み pollInterval。
lastPollInterval	最後に完了したポーリング周期の所要時間 (秒単位)。
timeSinceLastPoll	スナップショットが作成された時点での、最後のポーリングからの秒数。
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。
domainName	domainMgrId 列によって識別されるドメイン名。

discoveredIps ビュー

discoveredIps ビューは、NCIM トポロジー・データベース内のすべての IP アドレスを、関連するデバイスの詳細とともにリストします。

以下の表に、discoveredIps ビューの各列をリストします。

表 79. *ncmonitor discoveredIps* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
mainNode	NCIM entityData テーブルに示されている、メイン・ノードの entityName。
mainNodeEntityId	NCIM entityData テーブルに示されている、メイン・ノードの entityId。
entityId	NCIM entityData テーブルに示されている、インターフェースの entityId。
class	NCIM classMembers および entityClass テーブルで定義される、メイン・ノードの entityClass。

表 79. *ncmonitor discoveredIps* ビュー (続き)

列名	説明
mainNodeMgdStatus	NCIM managedStatus テーブルに示されている、メイン・ノードの現在の管理状況。
entityMgdStatus	NCIM managedStatus テーブルに示されている、エンティティーの現在の管理状況。
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。
domainName	domainMgrId 列によって識別されるドメイン名。

managementInterfaceIps ビュー

managementInterfaceIps ビューには、Network Manager が SNMP アクセス権を取得したすべてのデバイスの SNMP 管理インターフェース IP アドレスがリストされます。SNMP アクセス権が取得されなかったシャーシの IP アドレスはリストされません。

SNMP アクセスが可能なデバイスの場合、Network Manager によってシャーシに割り当てられる IP アドレスは、そのデバイス上のインターフェースの IP アドレスでもあることが必要です。したがってこのビューには、シャーシ ping ポーリングとインターフェース ping ポーリングの両方でモニターできる一連の IP アドレスが表示されます。

以下の表で、managementInterfaceIps ビューの各列について説明します。

表 80. *ncmonitor managementInterfaceIps* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
mainNode	NCIM entityData テーブルに示されている、メイン・ノードの entityName。
mainNodeEntityId	NCIM entityData テーブルに示されている、メイン・ノードの entityId。
entityId	NCIM entityData テーブルに示されている、インターフェースの entityId。
class	NCIM classMembers および entityClass テーブルで定義される、メイン・ノードの entityClass。
ifIndex	NCIM インターフェース・テーブルからのインターフェースの ifIndex。
mainNodeMgdStatus	最後のポーリング・プログラム・スナップショットの mainNodeEntityId 列で識別されるシャーシ・エンティティーの管理状況。
interfaceMgdStatus	最後のポーリング・プログラム・スナップショットの mainNodeEntityId 列で識別されるインターフェース・エンティティーの管理状況。
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。

表 80. *ncmonitor managementInterfaceIps* ビュー (続き)

列名	説明
domainName	domainMgrId 列によって識別されるドメイン名。

chassisOnlyIps ビュー

chassisOnlyIps ビューには、デバイスで IP アドレスを持つインターフェースがディスカバーされなかったために、シャーシ ping ポーリングでしかモニターできない IP アドレスがリストされます。通常これは、Network Manager がデバイスへの SNMP アクセスに失敗した場合ですが、ディスカバリー構成による場合もあります。

以下の表に、chassisOnlyIps ビューの列をリストします。

表 81. *ncmonitor chassisOnlyIps* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
mainNode	NCIM entityData テーブルに示されている、メイン・ノードの entityName。
mainNodeEntityId	NCIM entityData テーブルに示されている、メイン・ノードの entityId。
class	NCIM classMembers および entityClass テーブルで定義される、メイン・ノードの entityClass。
mainNodeMgdStatus	最後のポーリング・プログラム・スナップショットの mainNodeEntityId 列で識別されるシャーシ・エンティティーの管理状況。
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。
domainName	domainMgrId 列によって識別されるドメイン名。

unpollableIps ビュー

unpollableIps ビューは、ポーリング・プログラムが ping ポーリングを実行しない IP アドレスをリストします。これらの IP アドレスは、SNMP ポーリング・ポリシーを使用してモニターできます。

これらは、単一のインターフェースに複数の IP アドレスがあるマルチネット・インターフェースの 2 次 IP アドレスです。ping ポーリングは、NCIM シャーシ・テーブルおよびインターフェース・テーブルの accessIPAddress フィールドから作動するため、ping ポーリングを使用してモニターできるのはインターフェースごとに 1 つの IP アドレスだけです。

以下の表に、unpollableIps ビューの各列をリストします。

表 82. *ncmonitor unpollableIps* ビュー

列名	説明
ip	ディスカバーされて NCIM トポロジー・データベースに追加されたことが予想される、ドット表記の IPv4 アドレス。
mainNode	NCIM entityData テーブルに示されている、メイン・ノードの entityName。
mainNodeEntityId	NCIM entityData テーブルに示されている、メイン・ノードの entityId。
class	NCIM classMembers および entityClass テーブルで定義される、メイン・ノードの entityClass。
ifIndex	NCIM インターフェース・テーブルからのインターフェースの ifIndex。
interfaceAccessIp	この IP を 2 次 IP アドレスに持つ、マルチネット・インターフェースのポーリング可能なプライマリー IP アドレス。
mainNodeMgdStatus	最後のポーリング・プログラム・スナップショットの mainNodeEntityId 列で識別されるシャーシ・エンティティーの管理状況。
interfaceMgdStatus	最後のポーリング・プログラム・スナップショットの mainNodeEntityId で識別されるインターフェース・エンティティーの管理状況。
domainMgrId	NCIM トポロジー・データベースの domainMgr テーブルに示されている、関連するドメインの ID。
domainName	domainMgrId 列によって識別されるドメイン名。

OQL データベース (OQL databases)

組み込み OQL ポーリング・データベースには、複数のポーリング構成オプションが用意されています。

ポーリング用の構成データベース

構成データベースは、ポーリング・エンジン ncp_poller によってさまざまな目的で使用されます (診断とデバッグ、高可用性デプロイメントでのフェイルオーバーの支援、MIB グラファーのデバッグ、履歴パフォーマンス・データの保管制限の構成など)。

ポーリング用の構成データベースは、`$NCHOME/etc/precision/NcPollerSchema.cfg` で定義されています。

構成データベースには、以下のテーブルがあります。

- config.properties
- config.failover
- config.realTimeControl
- config.pruning

config.properties 表

config.properties 表では、複数のポーリング設定を構成することができます。

config.properties 表で値を設定するには、ファイル `$NCHOME/etc/precision/NcPollerSchema.cfg` を編集します。

次の表で、config.properties 表の列について説明します。

表 *83. config.properties* データベース表

列名	制約	データ型	説明
PolicyUpdateInterval		整数	ncp_poller プロセスが ncmonitor データベースでポーリング・ポリシー構成の変更をスキャンする間隔 (秒単位)。デフォルトは 30 秒です。
ManagedStatusUpdateInterval		整数	ncp_poller プロセスが NCIM managedStatus 表で変更をスキャンする間隔 (秒単位)。これは、ネットワーク・ビュー、ネットワーク・ホップ・ビュー、構造ブラウザーのいずれかの GUI で行われた管理状況の変更にポーリング・プログラムが対応するために要する最大時間です。デフォルトは 30 秒です。
LogAccessCredentials		Boolean Integer	SNMP アクセス資格情報 (コミュニティー・ストリングとパスワード) を、プレーン・テキストで表示するかどうかを制御します。False に設定すると、これらのストリングがアスタリスクから成る固定長のストリングで置き換えられます。デフォルトは False です。
UseGetBulk		Boolean Integer	デフォルトでは GetBulk は使用されません。このパラメーターは以下のいずれかの値に設定します。 • 0: デフォルト値。SNMP ヘルパーは GetBulk を使用しません。 • 1: SNMPv2 または v3 が使用される場合に、SNMP ヘルパーが GetNext 要求の代わりに GetBulk 要求を使用するように構成するには、この値を設定します。

表 83. *config.properties* データベース表 (続き)

列名	制約	データ型	説明
DefaultGetBulkMaxReps		整数	このプロパティーは、Network Manager プロセスにより発行される GetBulk 要求内の `max-repetitions` フィールドに割り当てる数値を定義します。GetBulk 要求に 1 つの varbind が含まれる場合は、値 20 が使用されます。複数の varbind が含まれる場合は、値が適切に調整 (varbind の数で除算) されるため、応答には、常にほぼ同数の varbind が含まれるようになります。

config.failover テーブル

config.failover では、ポーリングのフェイルオーバー設定を構成することができます。

以下の表で、config.failover テーブルの各列について説明します。

表 84. *config.failover* データベース表

列名	制約	データ型	説明
FailedOver	NULL 以外	Boolean Integer	高可用性デプロイメントでのフェイルオーバーを容易にするために使用されます。この値は変更してはなりません。この値をチェックして、システムがフェイルオーバーしたかどうかを判別できます。このフィールドは、以下のいずれかの値になります。 • 0: プライマリー・ドメインのポーリング・プログラムがアクティブにポーリングを行い、バックアップ・ポーリング・プログラムがスタンバイしています。 • 1: プライマリー・ポーリング・プログラムはアクティブにポーリングしておらず、バックアップ・ポーリング・プログラムが引き継いでいます。

config.realTimeControl 表

config.realTimeControl では、MIB グラファーでリアルタイムのポーリング・ポリシーを管理するための設定を構成することができます。

次の表で、config.realTimeControl 表の各列について説明します。この表は、MIB グラファー・アプリケーションがリアルタイムのポーリング・ポリシーを維持するために使用します。この表は一般には使用されませんが、問題が発生した場合に MIB グラフをデバッグするために使用できます。

表 *85. config.realTimeControl* データベース表

列名	制約	データ型	説明
POLICYID	NULL 以外 基本キー	整数	アクティブなリアルタイム・グラフが存在する場合は、グラフごとに作成され、この POLICYID フィールドを使用して参照されるポーリング・ポリシーに対応するレコードが、この表でグラフごとに作成されます。
HEARTBEATCOUNT	NULL 以外	整数	グラフがアクティブとなっていた期間を判定する目安となります。この値は、グラフがレコードを更新した回数を表します。
CHANGETIME		タイム・スタンプ	「ハートビート」が最後に受信されたタイミングを示す UNIX タイム・スタンプ。

config.pruning 表

config.pruning 表では、履歴パフォーマンス・データの保管制限を構成することができます。この表は、ポーリング・エンジンが ncpolldata pollData 表の行数に対する制限を構成する際に使用されます。

次の表で、config.pruning 表の各列について説明します。

表 *86. config.pruning* データベース表

列名	制約	データ型	説明
MAXPOLLDATAROWS	NULL 以外	Long 64 ビット	ncpolldata.polldata 表の最大許容行数を定義します。行数がこの限度を超えると、再び限度近くになるまで古いデータが除去されます。この数値を引き上げると、ヒストリカル・データに必要なストレージ・サイズが増えます。 **注意:** この値を引き上げると、このデータを使用するレポートのパフォーマンスが低下することもあります。

ポーリング用のプロファイル・データベース

プロファイル・データベースは、ポーリング・エンジンである ncp_poller がさまざまな目的で使用します (ポーリング・ポリシーとポリシー定義に関する要約情報、ping と SNMP 応答の統計、および一般的なプロファイル統計の保管など)。

ポーリング用のプロファイル・データベースは、`$NCHOME/etc/precision/NcPollerSchema.cfg` で定義されています。

プロファイル・データベースには、以下のテーブルがあります。

- profiling.policy
- profiling.icmp
- profiling.snmp
- profiling.engine

profiling.policy 表

profiling.policy 表は、ポーリング・ポリシーとポーリング定義に関する要約情報を提供します。

次の表で、profiling.policy 表の各列について説明します。

表 *87. profiling.policy* データベース表

列名	制約	データ型	説明
AVGSCOPETIME	NULL 以外	整数	各ポーリングのスコープを評価するのにかかる平均時間 (最初のポーリングをカウントするのではない)。
FIRSTSCOPETIME	NULL 以外	整数	初めて評価されるポーリングのスコープでエンティティーのリストにかかる時間 (CPU クロック・ティック数)。
ENTITYCOUNT	NULL 以外	整数	このポーリング・ポリシーとポーリング定義の組み合わせでモニターされるエンティティーの数。
POLICYID	NULL 以外 基本キー	整数	ncmonitor.poll.policyId フィールドの値。
POLICYNAME	NULL 以外	テキスト	ncmonitor.policy.policyName フィールドの値。
SCOPETIME	NULL 以外	整数	評価されるポーリングのスコープ (初回を除く) でエンティティーのリストにかかる合計時間 (CPU クロック・ティック数)。
SCOPECOUNT	NULL 以外	整数	ポーリングのスコープが評価された回数。
TARGETCOUNT	NULL 以外	整数	このポーリング・ポリシーとポーリング定義の組み合わせでポーリングされるアドレスの数。

表 87. *profiling.policy* データベース表 (続き)

列名	制約	データ型	説明
TEMPLATEID	NULL 以外 基本キー	整数	ncmonitor.poll.templateId フィールドの値。

profiling.icmp 表

profiling.icmp 表は、ping の応答統計に関する情報を保管します。

次の表で、 profiling.icmp 表の各列について説明します。

表 88. *profiling.icmp* データベース表

列名	制約	データ型	説明
IPVERSION	NULL 以外	テキスト	IPv4、IPv6、またはすべてのバージョン。
TIMEOUTS	NULL 以外	整数	再生が受信されていない ICMP 要求の数。
PACKETSIN	NULL 以外	テキスト	受信された ICMP パケットの数。
ERRORSIN	NULL 以外	整数	受信された ICMP エラーの数。
PACKETSOUT	NULL 以外	整数	送信された ICMP パケットの数。
ERRORSOUT	NULL 以外	整数	ICMP パケット送信時に発生したエラーの総数。

profiling.snmp 表

profiling.snmp テーブルは、SNMP の応答統計に関する情報を保管します。

次の表で、profiling.snmp 表の各列について説明します。

表 89. *profiling.snmp* データベース表

列名	制約	データ型	説明
ATTRIBUTESIN	NULL 以外	整数	受信された SNMP エラーの総数。
BACKOFFS	NULL 以外	整数	指数バックオフが開始された総回数。
DROPS	NULL 以外	整数	受信された未処理のパケットの総数。
ERRORSOUT	NULL 以外	整数	受信された tooBig エラーの総数。
GETOPERATIONS	NULL 以外	テキスト	SNMP Get 操作が実行された回数。
GETBULKSOUT	NULL 以外	整数	送信された SNMP Get Bulk 要求の総数。
IPADDR	NULL 以外	テキスト	ターゲット・デバイスの管理 IP アドレス。
GETNEXTOUT	NULL 以外	整数	送信された SNMP Get Next 要求の総数。
GETSOUT	NULL 以外	整数	送信された SNMP Get 要求の総数。

表 89. profiling.snmp データベース表 (続き)

列名	制約	データ型	説明
NOSUCHNAMESIN	NULL 以外	整数	受信された noSuchName エラーの総数。
PACKETSIN	NULL 以外	整数	受信された SNMP パケットの総数 (エラーを含む)。
PACKETSOUT	NULL 以外	整数	送信された SNMP パケットの総数。
RETRIES	NULL 以外	整数	再試行の総数。
SETERRORSIN	NULL 以外	整数	Set 要求から受信されたエラーの数。
SETOPERATIONS	NULL 以外	整数	SNMP Set 操作が実行された回数。
SETSOUT	NULL 以外	整数	送信された Set 要求の総数。
TIMEOUTS	NULL 以外	整数	タイムアウトになった SNMP 操作の総数。
TOOBIGSIN	NULL 以外	整数	受信された tooBig エラーの数。
WALKOPERATIONS	NULL 以外	整数	SNMP Walk 操作が実行された回数。

profiling.engine 表

profiling.engine 表は、一般的なプロファイル統計情報を保管します。

次の表で、profiling.engine 表の各列について説明します。

表 90. profiling.engine データベース表

列名	制約	データ型	説明
STARTTIME	NULL 以外	タイム・スタンプ	プロファイルが開始された時刻。
LASTUPDATE	NULL 以外	タイム・スタンプ	プロファイル統計が最終更新された時刻。
THREADSINUSE	NULL 以外	整数	コア・ポーリング・エンジン内のアクティブなスレッドの数。
BATCHESQUEUED	NULL 以外	整数	実行する必要があるが、使用できるスレッドがないバッチの数。
AVGBATCHTIME	NULL 以外	整数	各作業バッチの平均処理時間 (ミリ秒単位)。

付録 H. イベントの強化のデータベース

ここでは、イベント・エンリッチおよびイベント・ゲートウェイ・プラグインに使用されるデータベースの構造について説明します。

ncp_g_event データベース

イベント・ゲートウェイ・データベースでは、イベント・ゲートウェイ ncp_g_event を有効にして、Network Manager と Tivoli Netcool/OMNIbus の間でデータを転送します。

ncp_g_event データベースには、config というデータベース・スキーマがあります。

ほとんどのシステムでは、ゲートウェイのデフォルト構成が使用されています。構成設定を調整するには、イベント・ゲートウェイ config データベースに挿入されている値を変更します。このデータベースには、イベント・ゲートウェイの動作を定義する構成設定が含まれています。例えば、Network Manager と Tivoli Netcool/OMNIbus の間で使用されているマッピングおよび処理されるイベントを決定するフィルターを変更します。

イベント・ゲートウェイで使用されるエンティティー・データは、NCIM トポロジー・データベースのコピーである NCIM キャッシュに保管されます。NCIM キャッシュについて詳しくは、「*IBM Tivoli Network Manager IP Edition* トポロジー・データベース・リファレンス」を参照してください。

ncp_g_event コマンド行オプションの詳細については、*IBM Tivoli Network Manager IP Edition* 管理ガイド を参照してください。

構成データベース・スキーマ

構成データベースは、Tivoli Netcool/OMNIbus とNetwork Manager の間のイベント・マッピングを構成するために使用されます。

また、構成データベースは、Tivoli Netcool/OMNIbus と Network Manager の間で渡されるイベントの数を制限するフィルターを定義するためにも使用されます。

以下の表は、構成データベース・スキーマを要約しています。このスキーマは、`NCHOME/etc/precision/EventGatewaySchema.cfg` で定義されます。このファイルのドメイン固有のバージョンは、`NCHOME/etc/precision/EventGatewaySchema`*`domain_name`*`.cfg` というフォーマットを使用して指定できます。ここで、*domain_name* は使用しているドメインの名前です (例えば、`NCHOME/etc/precision/EventGatewaySchemaNCOMS.cfg`)。

表 *91.* 構成データベースの要約

データベース名	`config`
定義されている場所	`NCHOME/etc/precision/EventGatewaySchema.cfg`

© Copyright IBM Corp. 2006, 2016

表 91. 構成データベースの要約 (続き)

完全修飾データベース表名	config.defaults config.eventMaps config.failover config.nco2ncp config.ncp2nco config.precedence

以下のトピックでは、構成データベースのデータベース表について説明しています。

config.defaults テーブル

config.defaults テーブルには、イベント・ゲートウェイに関する一般構成データが入っています。

以下の表は、config.defaults テーブルについて説明しています。

注: フィールド NcoAuthUserName および NcoAuthPassword は、$NCHOME/etc/precision/NcoLogins.DOMAIN.cfg ファイルで構成されるようになりました。

表 92. *config.defaults* テーブルの説明

列名	制約	データ・タイプ (Data Type)	説明
IDUCFlushTime	NULL 以外	整数	ObjectServer からの Insert Delete Update Control (IDUC) フラッシュの間隔を秒単位で指定します。デフォルト値は 5 です。
ObjectServerUpdate Interval	NULL 以外	整数	イベント・ゲートウェイが、ObjectServer に対するイベント・エンリッチの更新をキューに入れる間隔を指定します。
NcpServerEntity	NULL 以外	テキスト	ポーリング・ステーションの IP アドレスを指定します。デフォルトでは、ゲートウェイは、Network Manager IP Edition のポーリング・ステーションがローカル・ホストで実行していると想定します。別のポーリング・ステーションを設定する場合、ポーリング・ステーションの IP アドレスを NcpServerEntityy フィールドに指定します。**注:** NcpServerEntity で指定されたデバイスが存在せず、トポロジー内で接続されていない場合、根本原因分析 (RCA) は分離抑止を実行できません。

関連タスク:

185 ページの『ObjectServer 更新間隔フィールドの構成』
イベント・ゲートウェイが ObjectServer に対するイベント・エンリッチの更新をキ

ューに入れる際の間隔を構成することができます。

201 ページの『ポーリング・プログラム・エンティティーの構成』
Network Manager サーバーがご使用のネットワーク・ドメインのスコープ内にない場合に RCA プラグインが分離抑止を実行できるようにするには、入口インターフェースの IP アドレスまたは DNS 名をポーリング・プログラム・エンティティーとして指定します。

config.precedence テーブル

config.precedence テーブルは、イベントをイベント ID 別にリストしており、複数のイベントが同じインターフェース上で発生するときに、優先するイベントを判別するのに必要な情報が含まれています。また、config.precedence テーブルは、イベント ID に基づいて、Tivoli Netcool/OMNIbus からのイベントを処理するために使用するイベント・マップを判別します。

以下の表は、config.precedence テーブルについて説明しています。

表 *93. config.precedence* テーブルの説明

列名	制約	データ・タイプ **(Data Type)**	説明
Precedence	NULL 以外	整数	ネットワーク・トポロジー内の同じエンティティーに対して複数のイベントがあるときに、根本原因分析 (RCA) プラグインによって使用される数を指定します。この数は優先されるイベントを判別するために使用されるため、インターフェース上の他のイベントを抑止します。リンクダウン・イベントの優先順位値が ping 失敗イベントの優先順位値よりも大きい場合、リンクダウン・イベントはインターフェース上の ping 失敗イベントを抑止します。 これらの優先順位値は固有です。 • 0 - この優先順位値を持つイベントはその他のイベントを抑止できません。このイベントは根本原因イベントになることはできません。優先順位値が 0 に設定される場合、イベントは症状イベントになるか、またはこのイベントに原因不明のマークを付けることができます。 • 10000 以上 - 10000 以上の優先順位値を持つイベントは抑止できません。イベントは症状イベントになることはできません。イベントは根本原因イベントにのみなるか、またはこのイベントに原因不明のマークを付けることができます。
EventMapName	NULL 以外	テキスト	一致する EventId を持つイベントを処理するために使用される config.eventMaps テーブルから、イベント・マップの名前を指定します。

表 93. *config.precedence* テーブルの説明 (続き)

列名	制約	データ・タイプ (Data Type)	説明
NcoEventId	基本キー NULL 以外	テキスト	alerts.status テーブルの EventId から、このテーブルで定義されている Precedence および EventMapName の値へのマッピングを指定します。 **注:** イベントがこのテーブルにリストされていない場合、イベントは generic-ip イベント・マップによって処理されます。

関連概念:

156 ページの『優先順位値』
イベントを処理するためのイベント・マップを選択すると同時に、数値の優先順位値がイベントに関連付けられます。この優先順位値は、同じエンティティーに対して複数のイベントがある場合に RCA プラグインによって使用されます。 エンティティーで最高の優先順位値を持つイベントは、他のイベントの抑止に使用されます。

config.eventMaps テーブル

config.eventMaps テーブルには、イベントの処理方法を示すイベント・マップが含まれます。テーブルは、イベント・ゲートウェイによって処理される Tivoli Netcool/OMNIbus イベントのそれぞれのタイプに固有の情報を保持します。

以下の表は、config.eventMaps テーブルについて説明しています。

表 94. *config.eventMaps* テーブルの説明

列名	制約	データ・タイプ (Data Type)	説明
EventMapName	基本キー NULL 以外	テキスト	イベント・マップの名前を指定します。この値は config.precedence テーブルによって参照されます。
HandledBy		テキスト	PolledEntityStitcher に代わるものとして、後方互換性を確保するために提供されています。既存のゲートウェイ eventMap には冗長なものがありますが、それらを完全に除去してしまうと、アップグレードの問題が生じます。このフィールドにより、既存の eventMap を新しい eventMap にマップして、あたかもその eventMap でイベントが処理されたように動作させることができます。

表 94. *config.eventMaps* テーブルの説明 (続き)

列名	制約	データ・タイプ **(Data Type)**	説明
EventCanFlap		Boolean	イベントがフラップする可能性があるかどうかを示します。フラップとは、デバイスまたはインターフェースがネットワークに接続されて切断されるという動作が、短時間の間に繰り返し行われる状態を言います。これが起こると、同じデバイスまたはインターフェースについて、問題イベントとクリア・イベントを交互に繰り返し受け取る事態が生じます。EventCanFlap = 1 を設定すると、この状態が RCA プラグインに通知されます。 RCA プラグインは、これらのイベントに TimedEscalation = 1 を指定して mojo.events データベースに配置し、そのまま 30 秒待機します。30 秒経つと、RCA プラグイン・スティッチャーの 1 つが、存続時間が 30 秒以上で、かつ TimedEscalation = 1 が設定されているすべてのイベントを処理します。イベントの処理を 30 秒待つことで、イベントを生成したエンティティーが安定し、フラップしないようになります。

関連概念:

121 ページの『イベント・マップによるスティッチャーの選択』
ここでは、イベント・マップによって特定のイベント・ゲートウェイ・スティッチャーを呼び出せるようにイベント・ゲートウェイを構成する方法を説明します。

config.nco2ncp テーブル

config.nco2ncp テーブルは、Tivoli Netcool/OMNIbus からNetwork Manager に渡されるイベントをフィルタリングするために使用されます。

以下の表は、config.nco2ncp テーブルについて説明しています。

表 *95. config.nco2ncp* テーブルの説明

列名	制約	データ・タイプ **(Data Type)**	説明
EventFilter	NULL 以外	テキスト	イベントがイベント・ゲートウェイによって処理されなければならないことを示すフィルターを指定します。フィルターに一致するイベントが処理されます。 重要: 変更の結果がわかっているのでなければ、このフィルターを変更しないでください。このフィルターを変更できるのは、上級者のみです。
StandbyEvent Filter		テキスト	プライマリー・サーバーがダウンし、バックアップ・サーバーがアクティブである場合に使用されます。スタンバイ・フィルターによって、ItnmHealthCheck イベントのみがイベント・ゲートウェイの通過を許可されます。これらのイベントはフェイルオーバー・プラグインに渡され、プライマリー・モードにスイッチバックするようシステムに指示します。
FieldFilter	外部定義 vblist データ型	Object	イベント・ゲートウェイに渡す alerts.status フィールドのサブセットを指定します。フィールド・フィルターが空である場合は、すべての alerts.status フィールドが渡されます。このフィルターの目的は、渡されるフィールドを必要最小限に抑えて、処理の負荷を軽減することです。

ゲートウェイは、ObjectServer がイベントを IDUC を使用する挿入として送信するか、または更新として挿入するかに応じて、新規レコードを挿入するか、または既存のレコードを更新するかを決定します。

関連概念:

102 ページの『着信イベント・フィルター』
着信イベント・フィルターは、ObjectServer からのイベントを除去し、特定の基準を満たすイベントのみを通過させます。

105 ページの『スタンバイ・フィルター』
フェイルオーバーのデプロイメントでは、スタンバイ・フィルターがフェイルオーバー・ペアのバックアップ・ドメインで使用されます。これは、プライマリーがアクティブである場合はバックアップ・ドメイン、バックアップがアクティブである場合はプライマリー・ドメインを意味します。スタンバイ・フィルターでは、正常性検査 (ItnmHealthCheck) イベントのみが Event Gateway を通過することができます。これらのイベントはフェイルオーバー・プラグインに渡され、プライマリー・モードにスイッチバックするようシステムに指示します。フェイルオーバーの動作のために、スタンバイ・フィルターに変更を行う場合は、このフィルターが引き続き正常性検査イベントを受け入れるようにする必要があることに注意してください。

config.ncp2nco テーブル

config.ncp2nco テーブルは、Network Manager IP Edition からTivoli Netcool/OMNIbus に渡されるイベントをフィルタリングし、マップするために使用されます。

以下の表は、config.ncp2nco テーブルについて説明しています。

表 *96. config.ncp2nco* テーブルの説明

列名	制約	データ・タイプ **(Data Type)**	説明
FieldFilter	外部定義 vblist データ型	Object	イベント・ゲートウェイが更新できる ObjectServer フィールドのセットを指定します。

関連概念:

107 ページの『発信フィールド・フィルター』
発信フィールド・フィルターは、イベント・ゲートウェイによって更新される可能性のある ObjectServer フィールド・セットを定義します。

config.failover テーブル

config.failover テーブルには、イベント・ゲートウェイ・コンポーネントのフェイルオーバー構成および現在のフェイルオーバー状態が含まれます。

重要: config.failover テーブルの値を手動で変更しないでください。フェイルオーバー構成では、FailedOver フィールドは仮想ドメイン・プロセスによって変更されます。

以下の表は、config.failover テーブルについて説明しています。

表 97. *config.failover* テーブルの説明

列名	制約	データ・タイプ (Data Type)	説明
FailedOver	NULL 以外	Boolean	フェイルオーバー状態を指定します。 • 0 - フェイルオーバー状態ではありません • 1 - フェイルオーバー状態です

ncp_g_event プラグイン・データベース

イベント・ゲートウェイ・プラグイン・データベース表は、プラグインが処理データを保管するために使用します。

RCA プラグイン・データベース

RCA プラグイン・データベース表は、RCA プラグインが根本原因分析を実行できるようにします。

mojo.events イベント・データベース表

mojo データベースは、イベント・ゲートウェイによって根本原因分析用に送信されるすべてのイベント・レコードを保管します。データベースには mojo.events テーブルが含まれます。

mojo データベースは NCHOME/etc/precision/RCASchema.cfg で定義されます。

レコードの列名は、イベント相関メソッドを構成するときに、多くの条件付きフィルターで使用されます。

以下の表は、mojo.events テーブル内の列について説明しています。

表 98. *mojo.events* データベース表列の説明

列名	制約	データ型	説明
ChangeTime	TIMESTAMP NULL 以外	長整数	イベントが RCA プラグインによって最終更新された時刻を示します。
CreateTime	TIMESTAMP NULL 以外	長整数	イベントが RCA プラグインによって最初に検出された時刻を示します。
説明		テキスト	イベントのテキスト説明を指定します。
EntityType	NULL 以外	Int	値 1 または 8 は、シャーシ・デバイスであることを示します。
EventId		テキスト	イベントのタイプ (NmosPingFail など)。
FirstOccurrence	TIMESTAMP NULL 以外		Tivoli Netcool/OMNIbus によってイベントが最初に検出された時刻。 **注:** この値は、Tivoli Netcool/OMNIbus ではなくプローブによって設定されます。つまり、このフィールドは、値が設定された状態で ObjectServer に送信されます。Tivoli Netcool/OMNIbus がこのフィールドを操作することはありません。

表 98. *mojo.events* データベース表列の説明 (続き)

列名	制約	データ型	説明
IsIsolationPoint	NULL 以外	Int	以下の値をとることができます。 • 0 - いいえ • 1 - はい
IsLoopbackInterface	NULL 以外	Int	以下の値をとることができます。 • 0 - いいえ • 1 - はい
IsMasterEvent	NULL 以外	Int	以下の値をとることができます。 • 0 - エンティティー上のマスター・イベントではありません。 • 1 - エンティティー上のマスター・イベントです。 **注:** エンティティー上のマスター・イベントは、そのエンティティー上の他のすべてのイベント (存在する場合) を抑止します。
IsOrphan	NULL 以外	Int	以下の値をとることができます。 • 0 - いいえ • 1 - はい **注:** このフィールドは、根本原因イベントがそれ以降削除されている抑止イベントを RCA プラグインが再処理できるようにするために、内部で使用されます。
LastOccurrence	TIMESTAMP NULL 以外	長整数	Tivoli Netcool/OMNIbus によってイベントが最後に検出された時刻。 **注:** この値は、Tivoli Netcool/OMNIbus がイベントを受け取ったときに自身で設定します。
NmosCauseType	NULL 以外	Int	以下の値をとることができます。 • 0 - 不明 • 1 - 根本原因 • 2 - 副次 • 3 - 抑止することも抑止されることもない
NmosEntityId	NULL 以外	Int	イベントが発生したエンティティー。
NmosObjInst	NULL 以外	Int	イベントが発生したエンティティーに関連するシャーシのエンティティー ID。
NmosSerial	NULL 以外	Int	このイベントを抑止したイベントのシリアル番号。
Precedence		Int	値 1 から 10,000 は、同一のエンティティーに複数のイベントが存在する場合に、そのエンティティー上の他のイベントを抑止するために使用されるイベントを示します。優先順位の値が最も高いイベントが、他のイベントを抑止します。
RemoteNodeAlias		テキスト	リモート・ネットワーク・エンティティーのネットワーク・アドレス。

表 98. *mojo.events* データベース表列の説明 (続き)

列名	制約	データ型	説明
Serial	基本キー NULL 以外	Uint	Tivoli Netcool/OMNIbus でのこのイベントのシリアル番号。イベントおよび mojo.events のレコードを一意的に識別するために使用されます。
Severity	NULL 以外	Int	イベントの重大度。
状態	NULL 以外	Int	このイベントのイベント状態。
SuppressionState	NULL 以外	Int	このイベントの抑止状態。このフィールドは、以下のいずれかの値になります。 • 0 - 抑止なし • 1 - エンティティー抑止 • 2 - 包含抑止 • 3 - 接続抑止 • 4 - 分離抑止 • 5 - ピア抑止
SuppressionTime	TIMESTAMP NULL 以外	長整数	イベントが最後に抑止された時刻。
TimedEscalation	NULL 以外	Int	以下の値をとることができます。 • 0 - イベントを直ちに処理するよう RCA プラグインに通知します。 • 1 - イベントを 30 秒後に処理するよう RCA プラグインに通知します。通常この値は、フラップする可能性があるイベントによって設定されます。 • 2 - 以前に `TimedEscalation = 1` が設定され、それ以降に処理されているイベントに対して設定されます。

config.defaults データベース表

config.defaults データベース表は、RCA プラグイン・イベント・キューの構成データを保管します。

`config` データベースは `NCHOME/etc/precision/RCASchema.cfg` で定義されます。

以下の表は、config.defaults データベース表内の列について説明しています。

表 99. *config.defaults* データベース表列の説明

列名	制約	データ型	説明
RequeueableEventIds		テキスト	RCA プラグイン・キューが非常に大きくなった場合に、特定のタイプのイベントを再びキューに登録できることを指定します。これにより、キュー内に特定のイベント・タイプのイベントが常に 1 つしか存在しないようになります。

表 *99. config.defaults* データベース表列の説明 (続き)

列名	制約	データ型	説明
MaxAgeDifference	NULL 以外	テキスト	RCA プラグインを通過するイベント間の存続期間の最大差を指定します。存続期間の差がここに指定する値を上回るイベントは、相互に抑止し合うことはできません。 デフォルトでは、このオプションはオフ、つまり、0 に設定されます。これは、同じエンティティーのイベントは、それぞれの存続期間に関係なく相互に抑止し合うことを意味します。
HonourManagedStatus	整数	Boolean	RCA プラグインが、イベントの根本原因を推測する際にエンティティーの管理状況を使用するかどうかを指定します。この値を 1 に設定すると、非管理デバイスからのイベントが無視されます。

関連概念:

159 ページの『RCA および非管理状況』
ここでは、RCA プラグインが非管理状態 (保守状態とも言います) にあるデバイスからのイベントを処理する方法について説明します。

関連タスク:

202 ページの『イベントの存続期間の最大差の構成』
デフォルトでは、同じエンティティーのイベントは、それぞれの存続期間に関係なく相互に抑止し合います。同じエンティティーについて、今日受け取ったイベントは昨日受け取ったイベントを抑止することができます。これを変更するには、RCA プラグインを通過するイベント間の存続期間の最大差を指定します。存続期間の差がここに指定する値を上回るイベントは、相互に抑止し合うことはできません。

SAE プラグイン・データベース

SAE プラグイン・データベース表は、MPLS VPN や IP パスなどのサービスに対して、SAE プラグインがサービスに影響を与えるイベントを生成できるようにします。

以下の表は、構成データベース・スキーマを要約しています。

表 *100.* 構成データベースの要約

データベース名	config
定義されている場所	NCHOME/etc/precision/SaeSchema.cfg NCHOME/etc/precision/SaeIPPath.cfg NCHOME/etc/precision/SaeItnmService.cfg NCHOME/etc/precision/SaeMplsVpn.cfg
完全修飾データベース表名	config.serviceTypes

関連概念:

148 ページの『SAE プラグイン』
SAE プラグインは、MPLS VPN および IP パスのサービスに影響を与えるイベン

トを生成します。

config.serviceTypes テーブル

config.serviceTypes テーブルは、SAE プラグインの構成情報を保管します。

以下の表は、config.serviceTypes テーブル内の列について説明しています。

表 *101. config.serviceTypes* データベース表列の説明

列名	制約	データ型	説明
ServiceTypeName	基本キー NULL 以外	テキスト	サービスのタイプ (「MPLS VPN エッジ・サービス」や「IP パス」など) を表します。このストリングは、ObjectServer 内の SAE イベントの eventId フィールドに取り込まれ、ObjectServer 内の SAE イベントの Summary フィールドの一部にもなります。
CollectionEntityType	NULL 以外	整数	SAE が生成されるコレクションに対応するエンティティー・タイプを指定するために使用されます (17 (VPN)、34 (ITNM サービス)、80 (IP パス) など)。NCIM entityType テーブルに組み込まれている可能性があるエンティティー・タイプのリストについては、「*IBM Tivoli Network Manager Topology Database Guide*」を参照してください。
ConstraintFilter	(オプション)	テキスト	必要に応じて、コレクション・テーブル内の関心のある項目を制約するために使用されます。例えば networkVpn テーブルでは、Type = 'MPLS Core' が設定されている項目が除外されます。この場合、制約フィルターは、「networkVpn->VPNTYPE <> 'MPLS Core'」と指定されます。
CustomerNameField	(オプション)	テキスト	ObjectServer 内のイベントの Summary フィールドに追加する顧客名前ストリングの取得場所を指定するために使用されます。例えば、ServiceEntity レコードに Customer フィールドが含まれている場合は、この列を使用して ServiceEntity トポロジー・レコードから「ACME Inc」などのストリングを取得できます。この例では、「entityData->DESCRIPTION」という形式になります。

ncmonitor 内の ncp_g_event プラグイン・データベース表

ここでは、ncmonitor データベース内にあるイベント・ゲートウェイ構成表と、各表に含まれている情報の種類について説明します。 これらの表のほとんどは、イベント・ゲートウェイのプラグイン構成に関係します。

以下の表では、ncmonitor データベース内のイベント・ゲートウェイ・プラグイン構成表をリストし、各表の目的について説明します。

表 *102. ncmonitor* 内のイベント・ゲートウェイ・プラグイン構成表

表	説明
ncmonitor. gwPluginTypes	使用可能なプラグイン・ライブラリーをリストします。 概念は、ポーリング・プログラムのテンプレート/ポーリング定義に似ています。 この表には、適応ポーリング・プラグイン用の単一のエントリーが含まれている必要があります。
ncmonitor. gwPlugins	有効になっているプラグインをリストします。 概念はポーリング・プログラム・ポリシーに似ています。 この表には、適応ポーリング・プラグイン用の少数のエントリーが含まれている必要があります。
ncmonitor. gwPluginEventMaps	各プラグインに関連するイベント・マップを識別します。 プラグインには、リストされたイベント・マップで処理されるイベントのみが提供されます。
ncmonitor. gwPluginEventStates	各プラグインに関連するイベントのタイプを識別します。 プラグインには、リストされたイベント状態で処理されるイベントのみが提供されます。
ncmonitor. gwSchemaFiles	イベント・ゲートウェイによって読み取られる OQL スキーマ・ファイルをリストします。 これらのファイルは、リストされた順に読み取られるため、表を定義する EventGatewaySchema ファイルが最初にリストされている必要があります。 デフォルトでは、EventGatewaySchema ファイルおよび NcoGateInserts ファイルがリストされます。 この表の目的は、今後のデバイス・サポートのために、既存の NcoGateInserts ファイルと同じフォーマットで、追加の自己完結型スキーマ・ファイルを提供できるようにすることです。
ncmonitor. gwPluginConf	特定のプラグインに対してオプションの構成変数を定義できるようにします。 **注:** これは、通常の環境では変更されない値を対象としています。

関連概念:

142 ページの『適応ポーリング・プラグイン』
ここでは、プラグインの前提条件、適応ポーリング・プラグインが activeEvent テーブルのフィールドにデータを取り込む方法、およびこのプラグインに関連する構成の詳細について説明します。 activeEvent テーブルは、NCMONITOR スキーマ内にあります。

144 ページの『Disco プラグイン』
ここでは、このプラグインの基本的な操作、プラグインの前提条件、およびこのプラグインに関連する構成の詳細について説明します。

150 ページの『zNetView プラグイン』
ここでは、プラグインの前提条件、およびプラグインに関連する構成の詳細について説明します。

関連タスク:

191 ページの『プラグインの有効化と無効化』
プラグインを有効または無効にするには、`ncp_gwplugins.pl` スクリプトを使用します。プラグインごとにスクリプトを個別に実行します。

192 ページの『プラグイン情報のリスト』
イベント・ゲートウェイ・プラグインに関する情報をリストすることができます。 例えば、各プラグインがサブスクライブしているイベント・マップとイベント状態をリストすることができます。

193 ページの『イベント・マップ・サブスクリプションの変更』
プラグインがサブスクライブしているイベント・マップを変更することができます。例えば、新しいイベント・マップを追加し、そのイベント・マップによって処理されるイベントに対してシステムによる RCA を実行する場合は、そのイベント・マップを RCA プラグインのサブスクリプション・リストに追加する必要があります。

195 ページの『プラグイン構成パラメーターの設定』
`ncp_gwplugins.pl` スクリプトを使用して、イベント・ゲートウェイ・プラグインのオプションの構成パラメーターを設定することができます。

付録 I. Network Manager 用語集

ここでは、Network Manager 製品に関連する用語を理解するための情報を提供します。

Network Manager に関連する用語を、以下のリストで説明します。

AOC ファイル (AOC files)
アクティブ・オブジェクト・クラス・マネージャー (ncp_class) が、ディスカバリー後にネットワーク・デバイスを分類するために使用するファイル。デバイス分類は、オブジェクト ID およびその他のデバイス MIB パラメーターに対する一連のフィルターを使用して、AOC ファイルに定義されます。

アクティブ・オブジェクト・クラス (AOC) (active object class (AOC))
アクティブ・オブジェクト・クラス・マネージャー (ncp_class) が、ディスカバリー後にディスカバーされたデバイスを分類するために使用する、ネットワーク・デバイスの事前定義済み階層トポロジー内のエレメント。

エージェント (agent)
ディスカバリー・エージェントを参照してください。

クラス階層 (class hierarchy)
アクティブ・オブジェクト・クラス・マネージャー (ncp_class) が、ディスカバリー後にディスカバーしたデバイスを分類するために使用する、ネットワーク・デバイスの事前定義済み階層トポロジー。

構成ファイル (configuration files)
Network Manager の各プロセスには、プロセス・データベース内の値を設定することでプロセスの動作を制御するために使用される、1 つ以上の構成ファイルがある。 構成ファイルは、ドメイン固有にすることもできます。

ディスカバリー・エージェント (discovery agent)
ディスカバリーで実行され、ディスカバーされたデバイスから詳細情報を取得するコード。

ディスカバリー構成 GUI (Discovery Configuration GUI)
ディスカバリー・パラメーターを構成するために使用する GUI。

ディスカバリー・エンジン (ncp_disco) (Discovery engine (ncp_disco))
ネットワーク・ディスカバリーを実行する Network Manager プロセス。

ディスカバリー・フェーズ (discovery phase)
ネットワーク・ディスカバリーは 4 つのフェーズに分かれている (デバイスへの問い合わせ、アドレスの解決、接続のダウンロード、および接続の相関)。

ディスカバリー・シード (discovery seed)
ディスカバリーの開始点となる 1 つ以上のデバイス。

© Copyright IBM Corp. 2006, 2016

ディスカバリー・スコープ **(discovery scope)**
1 つ以上のサブネットおよびネットマスクで表されるディスカバリーの境界。

ディスカバリー状況 **GUI (Discovery Status GUI)**
実行中のディスカバリーを起動してモニターするために使用する GUI。

ディスカバリー・スティッチャー **(discovery stitcher)**
ディスカバリー・プロセス中に使用されるコード部分。 さまざまなディスカバリー・スティッチャーがあります。これらのスティッチャーは、2 つのタイプにグループ化できます。1 つのタイプは、ディスカバリーのデータ収集フェーズでデータベース間のデータ転送を行うデータ収集スティッチャー、もう 1 つは、データ処理フェーズでネットワーク・トポロジーを構築するデータ処理スティッチャーです。

ドメイン **(domain)**
ネットワーク・ドメインを参照してください。

エンティティー **(entity)**
トポロジー・データベースの概念。 Network Manager によってディスカバーされるデバイスとデバイス・コンポーネントはすべてエンティティーである。 VPN や VLAN などのデバイス・コレクション、および複雑な接続を形成する部分的なトポロジーも、エンティティーである。

イベントの強化 **(event enrichment)**
トポロジー情報をイベントに追加するプロセス。

イベント・ゲートウェイ **(ncp_g_event) (Event Gateway (ncp_g_event))**
イベント・エンリッチを実行する Network Manager プロセス。

イベント・ゲートウェイ・スティッチャー **(Event Gateway stitcher)**
イベント・エンリッチ・プロセスの一環としてトポロジー・ルックアップを実行するスティッチャー。

フェイルオーバー **(failover)**
ご使用の Network Manager 環境でフェイルオーバー・アーキテクチャーを使用することにより、高可用性を実現するシステムを構成し、コンピューターまたはネットワークの障害の影響を最小限に抑えることができます。

フェイルオーバー・プラグイン **(Failover plug-in)**
イベント・ゲートウェイから Network Manager 正常性検査イベントを受け取り、それらのイベントを仮想ドメイン・プロセスに渡します。仮想ドメイン・プロセスは、フェイルオーバーを開始するかどうかをそのイベントに基づいて決定します。

障害検出ビュー **(Fault Finding View)**
上部の **Active Event List (AEL)** ポートレットと下部の ネットワーク・ホップ・ビュー ポートレットからなる複合 GUI ビュー。 障害検出ビューを使用して、ネットワーク・イベントをモニターします。

フル・ディスカバリー **(full discovery)**
広いスコープで実行され、管理するネットワーク・デバイスのすべてをディスカバーすることを目的としているディスカバリー実行。 通常、フル・デ

ィスカバリーは、部分的なディスカバリーと対比する場合を除いて、単にディスカバリーと呼ばれます。 部分的なディスカバリーも参照してください。

メッセージ・ブローカー **(message broker)**
Network Manager プロセス間の通信を管理するコンポーネント。 Network Manager によって使用されるメッセージ・ブローカーは、Really Small Message Broker と呼ばれます。 Network Manager が正しく動作するようにするには、Really Small Message Broker を常時実行中にする必要があります。

NCIM データベース **(NCIM database)**
トポロジー・データ、ポーリング・ポリシーとポーリング定義に関連付けられているデータなどの管理データ、およびデバイスからのパフォーマンス・データを格納するリレーショナル・データベース。

ncp_disco
ディスカバリー・エンジン を参照してください。

ncp_g_event
イベント・ゲートウェイを参照してください。

ncp_model
トポロジー・マネージャーを参照してください。

ncp_poller
ポーリング・エンジンを参照してください。

ネットワーク・ドメイン **(network domain)**
ディスカバーおよび管理対象のネットワーク・エンティティーの集合。 Network Manager を一度インストールすれば、複数のネットワーク・ドメインを管理することができます。

ネットワーク・ヘルス・ビュー **(Network Health View)**
上部の ネットワーク・ビュー ポートレットと下部の **Active Event List (AEL)** ポートレットからなる複合 GUI ビュー。 ネットワーク・ヘルス・ビューを使用して、ネットワーク・デバイスのイベントを表示します。

ネットワーク・ホップ・ビュー **(Network Hop View)**
ネットワーク視覚化 GUI。 特定のデバイスのネットワークを検索したり、指定したネットワーク・デバイスを表示したりするには、ネットワーク・ホップ・ビューを使用します。ネットワーク・ホップ・ビューは、ネットワーク・トラブルシューティングの出発点としても使用できます。 以前はホップ・ビューとして呼ばれていました。

ネットワーク・ポーリング **GUI (Network Polling GUI)**
管理者 GUI。 ポーリング・ポリシーおよびポーリング定義を定義できるようにする。

ネットワーク・ビュー **(Network Views)**
ディスカバーされたネットワークのビューを階層構造で表示するネットワーク視覚化 GUI。 ネットワーク・ビューは、ディスカバリーの結果を表示し、ネットワーク問題のトラブルシューティングを行う場合に使用します。

OQL データベース (OQL databases)
Network Manager プロセスでは、OQL データベースに構成情報、管理情報、および操作情報が保管されます。

OQL 言語 (OQL language)
Network Manager での使用を目的に設計された、構造化照会言語 (SQL) のバージョン。 Network Manager プロセスは、OQL を使用してデータベースを作成したり、データベースとやり取りしたりします。

部分的ディスカバリー (partial discovery)
以前にディスカバーされたネットワークのセクションの再ディスカバリー。通常、ネットワークのセクションは、一定のアドレス範囲、単一デバイス、または 1 つのデバイス・グループで構成されるディスカバリー・スコープを使用して定義されます。 部分的なディスカバリーは、最後に実行されたフル・ディスカバリーの結果を利用し、ディスカバリー・エンジン (ncp_disco) が最後のフル・ディスカバリー以降に停止されていない場合にのみ実行できます。 フル・ディスカバリーも参照してください。

パス・ビュー (Path Views)
選択した 2 つのデバイス間のネットワーク・パスを構成するデバイスおよびリンクを表示するネットワーク視覚化 GUI。 ネットワーク・オペレーターがネットワーク・パスを視覚化するのに役立つように、新しいパス・ビューを作成するか、既存のパス・ビューを変更します。

パフォーマンス・データ (performance data)
パフォーマンス・データは、パフォーマンス・レポートを使用して収集できます。 このレポートでは、モニター・システムで診断のため収集したパフォーマンスの履歴データを表示できます。

ポーリング・エンジン (ncp_poller) (Polling engine (ncp_poller))
ターゲットのデバイスとインターフェースに対してポーリングを行う Network Manager プロセス。 ポーリング・エンジンは、ポーリング対象のデバイスからパフォーマンス・データの収集も行う。

ポーリング定義 (poll definition)
ネットワーク・デバイスやインターフェースをポーリングする方法を定義し、さらにターゲットのデバイスやインターフェースをフィルター処理する。

ポーリング・ポリシー (poll policy)
ポーリングするデバイスを定義する。 ポーリングのその他の属性 (ポーリング頻度など) も定義する。

Tivoli Netcool/OMNIbus のプローブ (nco_p_ncpmonitor) (Probe for Tivoli Netcool/OMNIbus (nco_p_ncpmonitor))
Network Manager のポーリングおよびプロセスによって生成されたイベントを取得して処理し、これらのイベントを ObjectServer に転送します。

RCA プラグイン (RCA plug-in)
イベント内のデータおよびディスカバーされたトポロジーに基づき、RCA スティッチャー内でコーディングされているルールを使用して、他のイベントによって引き起こされるイベント、または他のイベントを引き起こすイベントを特定しようとします。

RCA スティッチャー (RCA stitcher)
トリガー・イベントが RCA プラグインを通過する間にこれを処理するスティッチャー。

根本原因分析 (RCA) (root-cause analysis (RCA))
1 つ以上のデバイス・アラートの根本原因を判別するプロセス。

SNMP MIB ブラウザー (SNMP MIB Browser)
ネットワーク問題の診断をサポートするために、ネットワーク・デバイスから MIB 変数情報を取得する GUI。

SNMP MIB グラファー (SNMP MIB Grapher)
デバイスの MIB 変数のリアルタイム・グラフを表示して、そのグラフを障害分析やネットワークの問題の解決に使用する GUI。

スティッチャー (stitcher)
ディスカバリー、イベント・エンリッチ、およびルート原因分析のプロセスで使用されるコード。 ディスカバリー・スティッチャー、イベント・ゲートウェイ・スティッチャー、およびRCAスティッチャーも参照してください。

構造ブラウザー (Structure Browser)
ネットワーク・デバイス内の障害を切り分けるために、デバイス・コンポーネントの正常性を調査できるようにする GUI。

トポロジー・マネージャー (ncp_model) (Topology Manager (ncp_model))
ディスカバリー後にトポロジー・データを保管し、そのトポロジー・データを、SQL を使用してトポロジー・データを照会できる NCIM トポロジー・データベースに送信します。

WebTools
ネットワーク・デバイスからデータを取得する、特化されたデータ取得ツール。ネットワーク視覚化 GUI。ネットワーク・ビュー、およびネットワーク・ホップ・ビューから起動できます。Web ブラウザーに URL を指定することでも起動できます。

特記事項

この情報は、IBM Tivoli Network Manager IP Edition 3.9 の PDF 文書セットに適用されます。

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で提供し、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。

© Copyright IBM Corp. 2006, 2016

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。

IBM Corporation
958/NH04
IBM Centre, St Leonards
601 Pacific Hwy
St Leonards, NSW, 2069
Australia
IBM Corporation
896471/H128B
76 Upper Ground
London
SE1 9PZ
United Kingdom
IBM Corporation
JBF1/SOM1 294
Route 100
Somers, NY, 10589-0100
United States of America

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が、その測定値が、一般に利用可能なシステムのもの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にお願いします。

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・システ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。

商標

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 Corporation 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、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Centrino、Intel Centrino ロゴ、Celeron、Xeon、Intel SpeedStep、Itanium、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。



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。

Linux は、Linus Torvalds の米国およびその他の国における登録商標です。

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。

Microsoft、Windows、Windows NT 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。

プライバシー・ポリシーに関する考慮事項

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品 (「ソフトウェア・オファリング」) では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、「ソフトウェア・オファリング」により個人情報が収集されることはありません。IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの

Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。

この「ソフトウェア・オファリング」は、Cookie もしくはその他のテクノロジーを使用して個人情報を収集することはありません。

このような目的での Cookie を含む様々なテクノロジーの使用の詳細については、IBM の『IBM オンラインでのプライバシー・ステートメント』(http://www.ibm.com/privacy/jp/ja/) を参照してください。

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]

アクセシビリティー xii
アップストリーム 166
アップストリーム・デバイス
  根本原因分析 166
イベント
  サービスに影響を与える (SAE)
    要約フィールド 197
  状況情報 242
  ネットワーク 242
  RCA の存続期間の最大差 202
イベント状態 110
イベント状態遷移図 112
イベント処理 114
イベント相関 99
イベントの強化 99
  イベント状態 110
  イベント状態遷移図 112
  イベント・エンリッチ・スティッチャー 133
  イベント・フィルター 102
  エンティティー取得スティッチャー 131
  クイック・リファレンス 99
  構成 181
  さまざまなイベント・タイプへの状態の割り当て 111
  着信イベント・フィルター 102
  着信フィールド・フィルター 106
  データ抽出スティッチャー 129
  デフォルトで使用されないスティッチャー 136
  トポロジー・ルックアップ・スティッチャー 124
  発信フィールド・フィルター 107
  フィールド・フィルター 106, 107
  優先順位値 156
イベント・エンリッチ 99
  イベント処理 114
  イベント・フィルター 102
  イベント・マップ 114
  インターフェース名によるイベントの強化 184
  スタンバイ・フィルター 105
イベント・エンリッチ (続き)
  スティッチャー 124
  追加のイベント・エンリッチの構成 181
  ポーリング・プログラム・エンティティー 166
  メイン・ノード・デバイスのロケーションによるイベントの強化 182
  例 137, 182, 184
イベント・エンリッチのデータベース 269
イベント・エンリッチ・スティッチャー 133
イベント・カテゴリー 241
イベント・ゲートウェイ
  イベント状態 110
  イベント状態遷移図 112
  イベント処理 114
  イベント・エンリッチ・スティッチャー 133
  イベント・マップ 114
  イベント・マップの選択 114
  イベント・マップの選択方法 115
  エンティティー取得スティッチャー 131
  構成 188
  さまざまなイベント・タイプへの状態の割り当て 111
  出力キュー 109
  スタンバイ・フィルター 105
  スティッチャー 121
  着信フィルター 102
  追加のイベント・エンリッチの構成 181
  データ抽出スティッチャー 129
  デフォルトで使用されないスティッチャー 136
  トポロジー・ルックアップ・スティッチャー 124
  フィルター 102
  プラグイン構成 表 280
  ポーリング・プログラム・エンティティー 158, 166
  優先順位値 156, 157
  NcpServerEntity 158
  OQL を使用したデータベースへのログイン 187
イベント・ゲートウェイ SAE プラグイン
  SAE タイプの追加 198
イベント・ゲートウェイのプラグイン情報リスト 192
イベント・ゲートウェイ・スティッチャー
  例 127, 130, 132, 134
イベント・ゲートウェイ・プラグイン 140
  構成 191
  サブスクリプション 193
  無効化 191
  有効化 191
  RCA 154
イベント・ゲートウェイ・プラグインの構成パラメーター
  設定 195
イベント・タイプ 111
イベント・フィールド 231
イベント・フィルター 102
  イベント・ゲートウェイへの着信 102
イベント・マッピング 272, 275
イベント・マップ 114
  イベント・ゲートウェイを使用する選択 114
  選択 114
  選択方法 115
  デフォルト 116
インターフェース
  アップストリーム 169
  組み込まれている 168
  ダウンストリーム 169
インターフェース ping ポーリング定義
  作成 32
  変更 48
演算子
  しきい値の式内 225
エンティティー
  カード 170
  仮想ルーター 170
  VLAN 170
エンティティー取得スティッチャー 131
オンライン資料 ix

[カ行]

カード
  entities 170
概要
  複数のポーリング・プログラム機能 79
仮想ルーター
  entities 170
環境変数、表記 xiv
漢字
  参照： マルチバイト・データ
管理
  適応ポーリング 61

© Copyright IBM Corp. 2006, 2016

管理 (続き)
ネットワーク・ポーリング 71
ヒストリカル・ポーリング・データ 84
複数のポーリング・プログラム機能 79
ポーリング 71
ポーリング・ポリシー・スロットル 75
管理状況
および RCA 159
規則、書体 xiv
基本しきい値の式
例 54
基本しきい値ポーリング
変更 37
基本しきい値ポーリング定義
作成 27
変更 44
基本しきい値ポリシー
デフォルト 206
キュー
イベント・ゲートウェイ上の出力 109
クイック・リファレンス
イベントの強化 99
RCA プラグイン 155
組み込まれているインターフェース 168
クリアのしきい値
例 219
検索
ポーリング・ポリシーの状況 72
研修
Tivoli 技術研修を参照 xiii
研修、Tivoli 技術 xiii
構成
イベントの強化 181
イベント・ゲートウェイ・プラグイン 191
根本原因の分析 201
追加のイベント・エンリッチ 181
ポーリング・プログラム・エンティティー 201
RCA プラグイン 201
SAE プラグイン 197
構成データベース 269
ポーリング用 262
コピー
ポーリング
ドメイン間の 73
get_policies.pl プログラム 73
コマンド
SIGHUP 188
根本原因の分析 154
構成 201
トポロジー・パスの検査 175
抑止のための存続期間の最大差 202
根本原因分析 168
組み込まれているインターフェース 168
根本原因分析 (続き)
ダウンストリーム・シャーシ・デバイス 171
分離抑止 171, 174
AMOS プロセス 167
config.defaults データベース 278
mojo.events データベース 276

[サ行]

削除
ポーリング定義 59
ポーリング・ポリシー 57
作成
インターフェース ping ポーリング定義 32
基本しきい値ポーリング定義 27
シャーシ ping ポーリング定義 32
適応ポーリング 67
汎用しきい値ポーリング定義 30
ポーリング定義 27
リモート ping ポーリング定義 34
リンク状態ポーリング定義 34
サブスクリプション
イベント・ゲートウェイ・プラグイン用 193
サポート情報 xiii
参照
イベントの強化 99
RCA プラグイン 155
シーケンス
RCA スティッチャー 162
しきい値の式
演算子 225
構文 221
例 54
eval ステートメントの使用 221
しきい値ポーリング
概要 9
例 9
シナリオ
適応ポーリング 61
シャーシ ping ポーリング定義
作成 32
変更 48
シャーシ・デバイス
ダウンストリーム 171
出力キュー
イベント・ゲートウェイ上 109
照会
OQL を使用した NCIM データベース 187
OQL を使用した ObjectServer 187
障害
シャーシ 169
状況情報イベント 242
状態
イベント 110
書体の規則 xiv
資料 ix
スコープ
ポーリング・ポリシーの 3
スタンバイ・フィルター 105
スティッチャー
イベント強化用 124, 133
イベント・ゲートウェイの例 127, 130, 132, 134
イベント・ゲートウェイ用 121
エンティティー取得用 131
データ抽出用 129
デフォルトで使用されないイベント・ゲートウェイ・スティッチャーの例 136
トポロジー・ルックアップ用 124
ExtractIfString.stch 130, 132
LookupIp.stch 127
RCA スティッチャーのシーケンス 162
RCA スティッチャーの説明 164
StandardEventEnrichment.stch 134
ストレージ容量
ストレージ限度の増加を判別するためのガイドライン 86
ストレージ限度の増加を判別するための考慮事項 84
例 87
スロットル
管理 75
接続されているインターフェース 169
接続しているエンティティー 170
設定
イベント・ゲートウェイ・プラグインの構成パラメーター 195
説明
RCA スティッチャー 164
選択
イベント・ゲートウェイを使用するイベント・マップ 114
イベント・マップ 114
選択方法
イベント・マップ用 115

[タ行]

ダウンストリーム 166
ダウンストリーム・シャーシ・デバイス
根本原因分析 171
ダウンストリーム・デバイス
根本原因分析 166
着信イベント・フィルター 102
着信フィールド・フィルター 106

追加のイベント・エンリッチ
構成 181
データ抽出スティッチャー 129
データベース 272, 275
イベント強化用 269
ゲートウェイ 269
構成 269
ポーリング用 249
config.defaults 270, 278
config.failover 275
config.nco2ncp 273
config.precedence 271
mojo.events 276
ncmonitor 249
ncp_g_event 269
データベース表
RCA プラグイン 276
SAE プラグイン 279
データ・ラベル 11
テーブル
expectedIps 252
pollLog 253
pollLogSummary 255
snmpAccess 250
snmpTarget 249
snmpUser 252
snmpv1Sec 251
snmpv3Sec 251
適応ポーリング
管理 61
作成 67
適応ポーリングのシナリオ 61
しきい値違反の確認 64
デバイスがダウンしたことの確認 61
適応ポーリング・プラグイン 142
デバイス
シャーシ 167, 169, 171
メイン・ノード 167
デフォルトのイベント・マップ 116
デフォルトの優先順位値 157
読者 vii
トポロジー・ルックアップ・スティッチャー 124
トリガーしきい値
例 219

[ナ行]

ネットワーク・イベント 242
ネットワーク・ポーリング
管理 71

[ハ行]

発信フィールド・フィルター 107
パラメーター
ポーリング定義の 5
ポーリング・ポリシー 2
汎用しきい値の式
例 54
汎用しきい値ポーリング
変更 37
例 219
汎用しきい値ポーリング定義
作成 30
変更 46
汎用しきい値ポリシー
デフォルト 206
ヒストリカル・ポーリング
データの削除 89
ヒストリカル・ポーリング・データ
管理 84
ビュー
chassisOnlyIps 261
delayedPollPolicies 258
discoveredIps 259
managementInterfaceIps 260
undiscoveredIps 256
unmanagedIps 257
unmonitoredIps 257
unpollableIps 261
unpolledFor15MinutesIps 258
フィールド・フィルター 106, 107
フィールド・マッピング
Network Manager から alerts.status への 234
フィルター
イベント・ゲートウェイへの着信 102
フェイルオーバー・プラグイン 146
複数のポーリング・プログラム機能
概要 79
管理 79
ポーリング・プログラムの除去 84
ポーリング・プログラムの追加 81
プラグイン 140
イベント・マップのサブスクリプション 152
プラグインの 152
構成 191
構成 表 280
サブスクリプション 193
適応ポーリング 142
フェイルオーバー (failover) 146
無効化 191
有効化 191
disco 144
PostNCIMProcessing 147
RCA 154
RCA のクイック・リファレンス 155
SAE 148
zNetView 150
プラグイン情報
リスト 192
プラグインの構成パラメーター
設定 195
プログラム
get_policies.pl 73
プロセス
イベント強化用 99
生成されたイベント 242
プロファイル・データベース
ポーリング用 266
分離抑止 171, 174
変更
インターフェース ping ポーリング定義 48
基本しきい値ポーリング定義 44
シャーシ ping ポーリング定義 48
汎用しきい値ポーリング定義 46
ポーリング定義 43
ポーリング・ポリシー 37
ポーリング・ポリシーのメンバーシップ・サイズをチェックする間隔 78
リモート ping ポーリング定義 50
リンク状態ポーリング定義 50
変数、表記 xiv
ポーリング
概要 1
管理 71
コピー 73
適応ポーリングの管理 61
適応ポーリングのシナリオ 61
デフォルト・プローブ 1
ドメイン間の 73
ネットワーク・ポーリングの管理 71
無効化 13
有効化 13
ポーリング状況テーブル 252
ポーリング定義 12
概要 4
削除 59
作成 27
しきい値 9
しきい値の式 221
データ・ラベル 11
定義 4
デフォルト 211
パラメーター 5
変更 43
例 52
メカニズム
ping 5
SNMP 6
リモート ping 8
SNMP リンク状態 6
ポーリング定義エディター 27
ポーリング定義タイプ 10

ポーリング定義の変更
例 52
ポーリングの無効化 13
ポーリングの有効化 13
ポーリング用データベース
config.failover テーブル 264
config.properties 表 263
config.pruning 表 265
config.realTimeControl 表 265
profiling.engine 表 268
profiling.icmp 表 267
profiling.policy 表 266
profiling.snmp 表 267
ポーリング・データベース 249
組み込み OQL データベース 262
構成データベース 262
プロファイル・データベース 266
config.failover テーブル 264
config.properties 表 263
config.pruning 表 265
config.realTimeControl 表 265
NCMONITOR 249
profiling.engine 表 268
profiling.icmp 表 267
profiling.policy 表 266
profiling.snmp 表 267
ポーリング・プログラム
除去 84
追加 81
ポーリング・プログラム・エンティティー 158, 166
構成 201
ポーリング・ポリシー
概要 2
削除 57
状況の取得 72
定義 2
デフォルト
基本しきい値 206
汎用しきい値 206
リモート ping 206
レポート作成 210
ping 205
パラメーター 2
変更 37
無効化 72
有効化 72
リフレッシュ 73
例 42
ポーリング・ポリシー (poll policy)
スコープ 3
ポーリング・ポリシーでのネットワーク・ビューの更新
無効化 78
有効化 78
ポーリング・ポリシーのネットワーク・ビューの更新
無効化 78
有効化 78
ポーリング・ポリシーのメンバーシップ・サイズ
チェックする間隔の変更 78
ポーリング・ポリシーのメンバーシップ・サイズをチェックする間隔
変更 78
ポーリング・ポリシー・スロットル
管理 75
ポリシー・スロットル
管理 75

［マ行］

マニュアル ix
マルチバイト・データ
ポーリング定義 12
マルチバイト・データを使用して定義 12
無効化
イベント・ゲートウェイ・プラグイン 191
ポーリング・ポリシー 72
ポーリング・ポリシーでのネットワーク・ビューの更新 78

［ヤ行］

有効化
イベント・ゲートウェイ・プラグイン 191
ポーリング・ポリシー 72
ポーリング・ポリシーでのネットワーク・ビューの更新 78
優先順位値 156
デフォルト 157
用語集 283

［ラ行］

リスト
イベント・ゲートウェイのプラグイン情報 192
リフレッシュ
ポーリング・ポリシー 73
リモート ping ポーリング
概要 8
制限 8
リモート ping ポーリング定義
作成 34
変更 50
リモート ping ポリシー
デフォルト 206
リンク状態ポーリング
変更 37
リンク状態ポーリング定義
作成 34
変更 50
ループバック・インターフェース 167
ルール・ファイル処理の例 229
ルックアップ・スティッチャー 124
例
イベント・エンリッチ 137, 182, 184
イベント・エンリッチ・スティッチャー 134
イベント・ゲートウェイ・スティッチャー (Event Gateway stitcher) 127, 130, 132, 134
エンティティー取得スティッチャー 132
しきい値の式 54
しきい値ポーリング 9
データ抽出スティッチャー 130
トポロジー・ルックアップ・スティッチャー 127
ポーリング定義
しきい値の式の例 54
レポート作成ポリシー
デフォルト 210
ロケール
参照： マルチバイト・データ

A

alerts.status テーブル
Network Manager で使用されるフィールド 234
AMOS プロセス
根本原因分析 167

C

chassisOnlyIps ビュー 261
config.defaults データベース 270
根本原因分析 278
config.eventMaps 272
config.eventMaps データベース 272
config.failover データベース 275
config.failover テーブル
ポーリング用 264
config.nco2ncp データベース 273
config.ncp2nco データベース 275
config.ncp2nco テーブル 275
config.precedence データベース 271
config.properties 表
ポーリング用 263
config.pruning 表
ポーリング用 265

config.realTimeControl 表
ポーリング用 265
config.serviceTypes テーブル 280

D

delayedPollPolicies ビュー 258
disco プラグイン 144
discoveredIps ビュー 259

E

EntityFromIfString.stch 132
eval ステートメント
しきい値の式内 221
expectedIps 表 252
ExtractIfString.stch 130

I

interfaces 172
ループバック 167

L

LookupIp.stch 127

M

managementInterfaceIps ビュー 260
mojo.events データベース
根本原因分析 276

N

NCIM データベース
OQL を使用した照会 187
NCMONITOR
ポーリング状況テーブル 252
ポーリング・データベース 249
ncmonitor データベース
イベント・ ゲートウェイ・プラグイン構成表 280
chassisOnlyIps ビュー 261
delayedPollPolicies ビュー 258
discoveredIps ビュー 259
expectedIps 表 252
managementInterfaceIps ビュー 260
pollLog 表 253
pollLogSummary 表 255
SNMP 249
snmpAccess テーブル 250
snmpTarget テーブル 249
snmpUser テーブル 252
ncmonitor データベース (続き)
snmpv1Sec テーブル 251
snmpv3Sec テーブル 251
undiscoveredIps ビュー 256
unmanagedIps ビュー 257
unmonitoredIps ビュー 257
unpollableIps ビュー 261
unpolledFor15MinutesIps ビュー 258
NcpServerEntity 158
ncp_g_event 269
Network Manager イベント・フィールド 231
Network Manager から alerts.status へのマッピング 234
Network Manager のイベント・カテゴリー 241
Network Manager 用語集 283

O

ObjectServer
OQL を使用した照会 187
OQL ポーリング・データベース 262

P

ping ポーリング
概要 5
変更 37
ポーリング定義タイプ 10
ping ポリシー
デフォルト 205
pollLog 表 253
pollLogSummary 表 255
PostNCIMProcessing プラグイン 147
profiling.engine 表
ポーリング用 268
profiling.icmp 表
ポーリング用 267
profiling.policy 表
ポーリング用 266
profiling.snmp 表
ポーリング用 267

R

RCA 154
および管理状況 159
構成 201
トポロジー・パスの検査 175
分離抑止 171, 174
優先順位値 156, 157
抑止のための存続期間の最大差 202
RCA スティッチャー 161
シーケンス 162
RCA スティッチャー (続き)
説明 164
RCA によって使用されるトポロジー・パスの検査 175
RCA の存続期間の最大差 202
RCA のトポロジー・パス 175
RCA のパス 175
RCA の例
関連した論理インターフェース 173
組み込まれているインターフェース 168
接続されているインターフェース 169
直接接続されたインターフェース 172
RCA プラグイン
クイック・リファレンス 155
構成 201
データベース表 276

S

SAE
要約フィールドの構成 197
SAE プラグイン 148
構成 197
データベース表 279
config.serviceTypes テーブル 280
SAE タイプの追加 198
SAE プラグインへの SAE タイプの追加 198
SIGHUP コマンド 188
SNMP
ncmonitor データベース 249
SNMP ポーリング
概要 6
ポーリング定義タイプ 10
SNMP リンク状態ポーリング
概要 6
snmpAccess テーブル 250
snmpTarget テーブル 249
snmpUser テーブル 252
snmpv1Sec テーブル 251
snmpv3Sec テーブル 251
StandardEventEnrichment.stch 134

T

Tivoli Netcool/OMNIbus のプローブ
構成 227
プロパティー・ファイル 227
ルール・ファイル 228
Tivoli 技術研修 xiii
Tivoli ソフトウェア情報センター ix

U

undiscoveredIps ビュー 256
unmanagedIps ビュー 257
unmonitoredIps ビュー 257
unpollableIps ビュー 261
unpolledFor15MinutesIps ビュー 258

V

VLAN
  entities 170

Z

zNetView プラグイン 150

［特殊文字］

<$nopage> 根本原因分析、RCA を参照
  ポーリング・プログラム・エンティティー 166
  例 166



Printed in the Republic of Ireland

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