

CASystemEDGE

ユーザ ガイド

リリース 5.7.1



このドキュメント（組み込みヘルプシステムおよび電子的に配布される資料を含む、以下「本ドキュメント」）は、お客様への情報提供のみを目的としたもので、日本 CA 株式会社（以下「CA」）により随時、変更または撤回されることがあります。

CA の事前の書面による承諾を受けずに本ドキュメントの全部または一部を複写、譲渡、開示、変更、複本することはできません。本ドキュメントは、CA が知的財産権を有する機密情報です。ユーザは本ドキュメントを開示したり、
(i) 本ドキュメントが関係する CA ソフトウェアの使用について CA とユーザとの間で別途締結される契約または (ii) CA とユーザとの間で別途締結される機密保持契約により許可された目的以外に、本ドキュメントを使用することはできません。

上記にかかわらず、本ドキュメントで言及されている CA ソフトウェア製品のライセンスを受けたユーザは、社内でユーザおよび従業員が使用する場合に限り、当該ソフトウェアに関連する本ドキュメントのコピーを妥当な部数だけ作成できます。ただし CA のすべての著作権表示およびその説明を当該複製に添付することを条件とします。

本ドキュメントを印刷するまたはコピーを作成する上記の権利は、当該ソフトウェアのライセンスが完全に有効となっている期間内に限定されます。いかなる理由であれ、上記のライセンスが終了した場合には、お客様は本ドキュメントの全部または一部と、それらを複製したコピーのすべてを破棄したことを、CA に文書で証明する責任を負います。

準拠法により認められる限り、CA は本ドキュメントを現状有姿のまま提供し、商品性、特定の使用目的に対する適合性、他者の権利に対して侵害のないことについて、黙示の保証も含めいかなる保証もしません。また、本ドキュメントの使用に起因して、逸失利益、投資損失、業務の中断、営業権の喪失、情報の喪失等、いかなる損害（直接損害か間接損害かを問いません）が発生しても、CA はお客様または第三者に対し責任を負いません。CA がかかる損害の発生の可能性について事前に明示に通告されていた場合も同様とします。

本ドキュメントで参照されているすべてのソフトウェア製品の使用には、該当するライセンス契約が適用され、当該ライセンス契約はこの通知の条件によっていかなる変更も行われません。

本ドキュメントの制作者は CA です。

「制限された権利」のもとでの提供: アメリカ合衆国政府が使用、複製、開示する場合は、FAR Sections 12.212、52.227-14 及び 52.227-19(c)(1)及び(2)、ならびに DFARS Section 252.227-7014(b)(3) または、これらの後継の条項に規定される該当する制限に従うものとします。

Copyright © 2013 CA. All rights reserved. 本書に記載された全ての製品名、サービス名、商号およびロゴは各社のそれぞれの商標またはサービスマークです。

CA Technologies 製品リファレンス

このマニュアルが参照している CA Technologies の製品は以下のとおりです。

- CA eHealth®
- CA Embedded Entitlements Manager (CA EEM)
- CA IT Asset Manager (CA ITAM)
- CA ITCM (CA IT Client Manager)
- CA NSM (CA Network and Systems Management)
- CA Patch Manager
- CA Server Automation
- CA Service Desk Manager (CA SDM)
- CA Spectrum®
- CA SystemEDGE
- インフラストラクチャ マネージャ用の CA システム性能
- CA Virtual Assurance for Infrastructure Managers
- CA Software Delivery

CA への連絡先

テクニカル サポートの詳細については、弊社テクニカル サポートの Web サイト (<http://www.ca.com/jp/support/>) をご覧ください。

目次

第 1 章: はじめに	17
CA SystemEDGE	17
対象読者.....	17
関連ドキュメント.....	17
規則	18
用語	20
 第 2 章: 機能および統合概要	 21
機能概要.....	21
モニタリング機能.....	22
状態管理.....	26
サポートされている MIB	27
MIB 拡張.....	32
修正処置.....	32
プラグインと統合	33
Application Insight Module (AIM)	34
CA Virtual Assurance	35
CA Server Automation.....	37
CA Spectrum.....	38
CA eHealth	39
CA NSM	40
 第 3 章: コンセプト	 41
アーキテクチャの概要.....	42
SNMP のバージョンとアクセス	43
SNMP トラップ	45
SNMP テーブル	45
インスタンスによるモニタの作成.....	47
設定ファイル.....	48
モニタリング アーキテクチャ	50
SNMPv2 の行ステータス	51
ステートフル モニタリングの仕組み	53
ステートレス モニタリングの仕組み	54
セルフ モニタリング	56

プロセスとサービスのモニタリング	63
プロセス グループ モニタリング	65
ログ ファイル モニタリング	66
Windows イベント モニタリング	67
履歴の収集	68
状態管理モデル	70
オブジェクト ステータス	71
オブジェクト 集計	72
スーパー 集計	74
状態管理と集計の仕組み	76
状態管理トラップ	78
状態管理の設定オプション	79
レガシー モードの操作	79
レガシー モニタ	80
モニタの例	80
CA Virtual Assurance との統合	81
プラットフォーム管理モジュール	81
リモート設定および展開	82
イベント エンジン	83
CA Virtual Assurance セキュリティ オプション	84
AIM アーキテクチャ	85
オンデマンド モジュール	85
AIM と CA Virtual Assurance の統合	86
SystemEDGE エージェントの使用のガイドライン	87

第 4 章: インストールと展開 91

インストールに関する注意事項	92
インストール時のエージェントの設定	92
CA Virtual Assurance を使用したインストール	93
Windows システムでのインストール	93
Windows でのエージェントのインストール	94
コマンドラインを使用した Windows へのエージェントのインストール	100
UNIX および Linux システムでのインストール	108
UNIX システムおよび LINUX システムでのエージェントのインストール	108
コマンドラインを使用した UNIX へのエージェントのインストール	115
\$CASYSEGE 変数のレガシー サポート	123
応答ファイルの設定と使用	125
レガシー モードでのエージェントのインストール	126
エージェントの展開	127

CA Virtual Assurance によるエージェントの展開	128
SystemEDGE のアンインストール	130
Windows 上の SystemEDGE および AIM のアンインストール	131
UNIX システム上の SystemEDGE および AIM のアンインストール	133
再インストール.....	135
管理対象ノードおよび AIM サーバのアップグレード.....	135
エージェントと AIM のアップグレード	136
ポリシーへの SystemEDGE モニタのインポート	139

第 5 章: エージェントの設定 141

設定の基本.....	141
sysedge.cf と sysedge.mon のやり取り	143
CA Virtual Assurance を使用した設定	143
ファイルベースの設定	144
システム情報の設定.....	145
SNMP 情報の設定.....	145
認証失敗トラップの設定.....	154
SNMP SET 通知の設定	155
SNMP SET 制限の設定	156
GetBulk 応答メッセージサイズの設定	157
連邦情報処理規格 (FIPS) モードの設定	158
診断ロギング メカニズムの設定	159
保守モードの設定.....	161
アクションのサポートの設定.....	162
サブプログラムのユーザおよびグループ権限の設定 (UNIX のみ)	162
MIB テーブルの制限の設定	163
デバイス ステータス確認の制限の設定	164
ウォーム スタート ディスカバリの設定	166
Linux の空きメモリ計算の設定	167
多層階層でのモニタ集計の設定.....	167
モニタリング設定.....	171
SystemEDGE AIM のロード.....	172
セキュリティの設定に関する推奨事項.....	174
モニタおよび自動ウォッチャーの正規表現の例	175
Perl 互換正規表現 (PCRE) のサポート.....	178
複数行の正規表現.....	180
テキスト パターン除外の設定	181
正規表現の例.....	181
SNMPv3 設定	185

SNMPv3 エンジン ID の設定	186
SNMPv3 ユーザ情報の設定	186
SNMPv2c および SNMPv3 のトラップ デスティネーションの設定	192
SNMPv3 設定ファイルの暗号化	195
SNMPv1 および SNMPv2c の無効化	195
Windows の SystemEDGE コントロール パネル	196

第 6 章: エージェントの起動 197

Windows システムでのエージェントの起動または停止	197
UNIX システムでのエージェントの起動または停止	198
UNIX システム用のサービス起動スクリプト	198
コマンドライン起動オプション	200
システム ブート時のエージェントの自動起動	201
Windows システムでのエージェントの自動起動	202
UNIX システムでのエージェントの自動起動	202
エージェントのウォーム スタート	202
ウォーム スタートの例外	203
保守モード	203

第 7 章: Systems Management Empire MIB 205

Systems Management Empire MIB の概要	206
システム グループ	207
デバイス テーブル	207
カーネル設定グループ	208
ブート設定グループ	208
ストリーム グループ	209
ユーザ テーブル	209
グループ テーブル	209
プロセス テーブル	209
プロセスの nice 値を変更する (UNIX のみ)	210
プロセスにシグナルを送信する	210
Who テーブル	210
リモート シェル グループ	211
リモート コマンドの実行	211
パフォーマンス グループ	212
プロセス間通信グループ	213
IPC メカニズムの削除	214
バッファ グループ	214

メッセージバッファ グループ	214
ストリーム バッファ グループ	215
I/O バッファ キャッシュ グループ	215
ディレクトリ名のルックアップ キャッシュ グループ	215
AIX 論理パーティション グループ	216
ワーク ロード マネージャ (WLM) クラスのテーブル	216
トラップ コミュニティ テーブル	216
NT システム グループ	217
NT スレッド テーブル	217
NT レジストリ グループ	217
NT サービス テーブル	218
NT パフォーマンス グループ	218
NT イベント モニタ グループ	219
NT レジストリ/パフォーマンス拡張グループ	220
セルフ モニタ テーブル	221
分散システム グループ	221
RPC 統計グループ	222
NFS 統計グループ	222
履歴テーブル	223
ログ モニタ テーブル	224
ディスク統計テーブル	224
AIX システムのディスク パフォーマンス統計収集の設定	225
CPU 統計テーブル	226
拡張グループ	227
プロセス モニタ テーブル	227
[プロセス グループ モニタ] テーブル	228
集計状態テーブル	228
LPAR グループ	228
ミラー テーブル	229
自動ウォッチャー テーブル	229
サポートされていない Systems Management Empire MIB オブジェクト	229

第 8 章: セルフ モニタリング 231

セルフ モニタリングの概要	231
セルフ モニタの状態管理	232
管理対象オブジェクトの作成	232
集計状態テーブル	234
セルフ モニタ テーブル	235
セルフ モニタ テーブルの列	236

サンプルセルフ モニタ テーブル エントリ	243
行作成.....	244
セルフ モニタ テーブルのフラグ	245
セルフ モニタ テーブルのアクション パラメータ	249
セルフ モニタ テーブルの表示.....	251
セルフ モニタリングの設定	252
monitor ディレクティブ -- セルフ モニタ テーブルへのエントリの追加	254
OID の逆ルックアップ	258
セルフ モニタリングの例.....	260
セルフ モニタリング用の edgemon コマンド.....	278
edgemon の例	282
セルフ モニタリング エントリの削除	283

第 9 章: プロセスとサービスのモニタリング 285

プロセスとサービスのモニタリングの概要	285
Windows サービスのモニタリング	286
プロセス モニタの状態管理.....	287
管理対象オブジェクトの作成.....	288
集計状態テーブル.....	289
プロセス モニタ テーブル	290
プロセス モニタ テーブルの列.....	291
サンプルプロセス モニタ テーブル エントリ	299
プロセス属性.....	301
行作成オブジェクト.....	303
プロセス モニタ テーブルのフラグ	305
プロセス モニタ テーブルのアクション パラメータ	312
プロセス モニタ テーブルの表示	315
プロセス モニタリングの設定	315
watch process ディレクティブ -- プロセス モニタ テーブルへのエントリの追加	317
watch ntservice ディレクティブ -- プロセス モニタ テーブルへのサービス モニタリング エン トリの追加.....	321
プロセス モニタリングの例.....	324
プロセス モニタリング用の edgemon コマンド.....	333
edgemon の例	336
プロセス モニタリング エントリの削除	340
Solaris ゾーン プロセスのモニタリング	341
プロセスとサービスのモニタリングの推奨事項	342

第 10 章: 自動ウォッチャー 343

自動ウォッチャーの仕組み	343
汎用自動ウォッチャー	345
プロセスとサービスの自動ウォッチャー	345
edgwatch ユーティリティ -- 自動ウォッチャーの使用	346
自動ウォッチャー用の edgwatch コマンド	351
edgwatch 自動ウォッチャーの例	356
sysedge.cf での自動ウォッチャーの指定	357
汎用自動ウォッチャーの例	361
サービス自動ウォッチャーの例	362
プロセス自動ウォッチャーの例	362

第 11 章: プロセス グループ モニタリング 363

プロセス グループ モニタリングの概要	363
[プロセス グループ モニタ] テーブル	364
[プロセス グループ モニタ] テーブルの列	365
行作成オブジェクト	371
プロセス グループ モニタ テーブルのフラグ	372
[プロセス グループ モニタ] テーブルのアクション パラメータ	374
[プロセス グループ モニタ] テーブルの表示	376
プロセス グループ モニタリングの設定	377
watch procgroup ディレクティブ -- [プロセス グループ モニタ] テーブルへのエントリの追加	378
watch procgroupex ディレクティブ -- [プロセス グループ モニタ] テーブルへのエントリの追加 (UNIX のみ)	380
プロセス グループ モニタリングの例	382
edgwatch ユーティリティ -- プロセス グループのモニタ	384
プロセス グループ モニタリング用の edgwatch コマンド	388
edgwatch の例	390
プロセス グループ モニタリング エントリの削除	390
Solaris ゾーン プロセス グループ モニタリング	391

第 12 章: ログ ファイル モニタリング 393

ログ ファイル モニタリングの概要	393
ディレクトリ モニタリング	394
ログ モニタ テーブル	394
ログ モニタ テーブルの列	395
行作成オブジェクト	399

ログ モニタ テーブルのフラグ	400
ログ モニタ テーブルのアクション パラメータ	404
ログ モニタ テーブルの表示	405
ログ モニタリングの設定	406
watch logfile ディレクティブ -- ログ モニタ テーブルへのエントリの追加	407
ログ ファイル モニタリングの例	409
edgewatch ユーティリティ -- ログ ファイルのモニタ	411
ログ ファイル モニタリング用の edgewatch コマンド	415
edgewatch の例	417
ログ モニタリング エントリの削除	419
ログ ファイル モニタリングの推奨事項	420
循環ログ ファイル	422

第 13 章: Windows イベント モニタリング 423

Windows イベント モニタリングの概要	423
Windows イベントの検索条件	424
NT イベント モニタ テーブル	425
NT イベント モニタ テーブルの列	425
行作成オブジェクト	429
NT イベント モニタ テーブルのフラグ	431
NT イベント モニタ テーブルのアクション パラメータ	434
NT イベント モニタ テーブルの表示	435
Windows イベント モニタリングの設定	436
watch ntevent ディレクティブ -- NT イベント モニタ テーブルへのエントリの追加	438
Windows イベント モニタリングの例	441
edgewatch ユーティリティ -- Windows イベントのモニタ	442
Windows イベント モニタリング用の edgewatch コマンド	447
edgewatch の例	449
Windows イベント モニタリング エントリの削除	451

第 14 章: 履歴の収集 453

履歴の収集の概要	453
履歴サンプリング	453
履歴制御テーブルと履歴データ テーブル	454
履歴制御テーブルの列	454
履歴テーブルの列	457
行作成オブジェクト	457
履歴サンプリングの例	458

履歴制御テーブルの表示.....	460
履歴制御テーブルの設定.....	460
起動時の初期設定.....	460
操作時の動的な設定.....	461
emphistory ディレクティブ -- 履歴制御テーブルへのエントリの追加.....	461
履歴の収集の例.....	462
履歴制御テーブルのエントリを管理するための emphistory コマンド.....	463

第 15 章: カスタム MIB オブジェクト 467

Systems Management Empire MIB 拡張グループ	467
拡張グループ機能.....	468
拡張変数の設定.....	469
拡張キーワード -- 拡張グループへのエントリの追加.....	470
追加パラメータ.....	471
拡張の例.....	471
拡張スクリプト.....	473
サンプル スクリプトのテスト.....	474
管理ソフトウェアでの拡張変数の使用.....	475
empire.asn1 で拡張変数に関する編集を行う方法.....	476
別個の MIB 仕様で拡張変数に関する編集を行う方法.....	476
拡張の使用に関する推奨事項.....	477

第 16 章: Windows レジストリとパフォーマンス MIB オブジェクト 479

Windows レジストリおよびパフォーマンス機能.....	479
Systems Management Empire MIB の ntRegPerf グループ	480
レジストリ データ.....	481
パフォーマンス データ.....	481
Windows レジストリおよびパフォーマンス変数の設定.....	483
ntregperf キーワード -- Windows レジストリとパフォーマンス MIB オブジェクトの追加.....	484
Windows レジストリとパフォーマンスの例.....	485
管理ソフトウェアでの Windows レジストリおよびパフォーマンス変数の使用.....	486
empire.asn1 で ntRegPerf 変数に関する編集を行う方法.....	486
ntRegPerf 変数に関する別個の MIB 仕様を追加する方法.....	487

第 17 章: SystemEDGE AIM 489

コア AIM.....	489
トッププロセス AIM	489
モニタ保守ウィンドウ AIM	490

モニタ保守ウィンドウ AIM の有効化	490
モニタ保守ウィンドウ AIM の制限事項	491
パフォーマンス キューブ AIM	491
パフォーマンス キューブ AIM の有効化.....	492

第 18 章: コマンドライン ユーティリティ 493

SystemEDGE コマンドライン ユーティリティ	494
SystemEDGE のその他のコマンドライン ユーティリティ	495
SystemEDGE コマンドライン ユーティリティの設定	495

第 19 章: 状態管理の使用例 497

Windows タスク スケジューラのモニタ	497
隔離ファイルの存在のモニタ	499
ファイル システムのモニタ	499
SNMP を介した拡張スクリプトの再設定	501
ログ ファイルのモニタと、電子メールによる通知の送信	505
モニタ プロセス	506
モニタ対象オブジェクトの作成	507

第 20 章: トラブルシューティングと使用方法に関する推奨事項 509

diagsysedge.exe の使用	509
diagsysedge によるエージェントが実行されているかどうかの確認.....	510
トラブルシューティング用のレポートの取得	511
AIM プラグインの検証	511
よくある問題と質問	512
トラブルシューティング用の診断ログ情報	513
エージェントが SNMP リクエストに応答しない	513
管理システムが SNMP トラップ メッセージを受信しない	516
バインド失敗 : アドレスがすでに使用中	518
CA SystemEDGE : xtrapmon.exe ユーティリティは edgetrapmon.exe に置き換え	518

付録 A: FIPS 140-2 の暗号化 519

FIPS 140-2 モード	519
FIPS ライブラリのインストール	519
プラットフォーム サポート	520
サポートされる暗号化プロトコル	520
サポートされる認証プロトコル	521

FIPS 140-2 モードの設定	521
FIPS モードの考慮事項	522
キー保護	522

付録 B: SystemEDGE Advanced Encryption 523

SystemEDGE Advanced Encryption	523
サポートされるプラットフォーム	523
サポートされる暗号化プロトコル	523
サポートされる認証プロトコル	524
FIPS の互換性	524
インストールの前提条件	524
SystemEDGE Advanced Encryption のインストール	524
Windows での SystemEDGE Advanced Encryption のインストール	525
Windows での SystemEDGE Advanced Encryption のアンインストール	525
UNIX での SystemEDGE Advanced Encryption のインストール	526
UNIX での SystemEDGE Advanced Encryption のアンインストール	526
インストールされるファイル	527

付録 C: 他の SNMP エージェントとの共存 529

複数の SNMP エージェントのサポート	529
複数の SystemEDGE インスタンスのサポート	530
Microsoft Windows SNMP エージェントとの共存	531
HP SNMP エージェントとの共存	531
AIX SNMP エージェントとの共存	532

付録 D: Host Resources MIB 533

Host Resources MIB の概要	534
ホスト リソース システム グループ	535
ホスト リソース ストレージ グループ	536
ホスト リソース デバイス グループ	536
デバイス テーブル	537
プロセッサ テーブル	538
ディスク ストレージ テーブル	538
パーティション テーブル	539
ファイル システム テーブル	540
ソフトウェア グループを実行するホスト リソース	541
ホスト リソースのインストール済みソフトウェア グループ	542
サポートされていない Host Resources MIB オブジェクト	543

付録 E: プライベート企業トラップ 545

トラップ PDU の形式	545
monitor トラップ	546
MonitorEntryNotReady トラップ	546
logMonMatch トラップ	547
logMonNotReady トラップ	547
ntEventMonMatch トラップ	548
ntEventMonNotReady トラップ	548
monitorClear トラップ	549
processStop トラップ	549
processStart トラップ	550
processThreshold トラップ	550
processClear トラップ	551
processNotReady トラップ	552
addrChange トラップ	552
procGroupChange トラップ	553
aggregateState トラップ	553
aggregateAdd トラップ	554
aggregateDelete トラップ	554
SNMPv1 トラップの形式	555

索引 557

第 1 章: はじめに

このセクションには、以下のトピックが含まれています。

[CA SystemEDGE](#) (P. 17)

CA SystemEDGE

SystemEDGE は、一元化された場所からネットワーク上のすべてのサーバを制御できるようにすることで、システム管理スタッフの生産性を高めます。また、システム管理タスクとインベントリ追跡を自動化し、増加するシステム サポート コストを削減しながら、生産性とシステムの安定性を向上させます。

SystemEDGE エージェントは、業界標準の SNMP による強力なシステム管理を提供します。このエージェントを使用すると、リモート管理システムから、システムの設定、ステータス、パフォーマンス、ユーザ、プロセス、ファイルシステムなどに関する重要な情報にアクセスできるようになります。また、エージェントにはインテリジェントなモニタリングおよび状態管理機能が含まれます。これにより、例外、オブジェクトの重大度の算出、ステータス情報をレポートおよび管理できます。

対象読者

このガイドは、UNIX、Linux および Windows サーバを管理するために SystemEDGE エージェントをインストール、設定、および使用する管理者を対象としています。使用するシステムのオペレーティングシステム環境、および SNMP (Simple Network Management Protocol、簡易ネットワーク管理プロトコル) を、ユーザが基本的に熟知していることを前提としています。

関連ドキュメント

CA Virtual Assurance のマニュアルは、次のマニュアルで構成されています。

管理ガイド

ユーザの環境の仮想リソースを管理するために、**CA Virtual Assurance** を管理および使用する方法を調査します。

インストール ガイド

簡単なアーキテクチャ情報、さまざまなインストール方法、インストール後の設定情報、および導入時の手順が含まれます。

オンライン ヘルプ

CA Virtual Assurance ユーザ インターフェースを使用するためのウィンドウの詳細、および手順の説明を提供します。

リファレンス ガイド

AutoShell、**CLI コマンド**、**MIB 属性**、およびパフォーマンス メトリックに関する詳細情報を提供します。

リリース ノート

オペレーティング システムのサポート、システム要件、発行済みの修正プログラム、各国語のサポート、既知の問題、およびドキュメントロードマップに関する情報を提供します。

サービス レスpons モニタリング ユーザ ガイド

SRM のインストールおよび設定の詳細が記載されています。

SystemEDGE ユーザ ガイド

SystemEDGE のインストールおよび設定の詳細について説明します。

SystemEDGE リリース ノート

オペレーティング システムのサポート、システム要件、および機能に関する情報を提供します。

規則

このガイドでは、以下の規則を使用します。

大文字と小文字の区別

このガイドで言及されるクラス、コマンド、ディレクティブ、環境パラメータ、関数、プロパティの名前はすべて大文字と小文字を区別します。また、記載されているとおりに正確に入力する必要があります。システム コマンドと環境変数名は、オペレーティング システムの要件に応じて、大文字と小文字が区別される場合があります。

相互参照

他のガイド、またはこのガイドの他のセクション内の情報への参照は、以下の形式で表示されます。

ガイド名

別のガイドの名前を示します。

「章名」

このガイドまたは別のガイドの章の名前を示します。

同義語

属性、オブジェクト、オブジェクト識別子（OID）などの用語は、このドキュメントでは「変数」と同義です。

SystemEDGE エージェント、CA SystemEDGE などの用語は、このドキュメントでは SystemEDGE と同義です。

構文

構文とユーザ入力では、以下の形式を使用します。

斜体

実際の値を指定する必要がある変数名またはプレースホルダを示します。

{a|b}

オペランド **a** または **b** を選択する必要があることを示します。

[] または [[]]

オプションのオペランドを示します。

構文例

以下の例は、上記の規則を使用しています。

```
modify -t ZONE [-m zoneserver] -p psetname {-min mincpu|-max maxcpu} pset -session ssh
```

オペランド **-min** および **-max** は必須ですが、プロセッサセット内の CPU の最小数または最大数のどちらを定義するかに応じて、いずれか 1 つのみを使用します。オペランド **-m** は、指定しなくてもこのコマンドは機能します。コマンドのほかの部分は記載されているとおりに入力する必要があります。

デフォルト ディレクトリ

パス ステートメント内で使用する **CASYSEEDGE** は、**SystemEDGE** がインストールされているディレクトリを示します。 デフォルト :

C:¥Program Files¥CA¥SystemEDGE.

インストール パス

パス ステートメント内で使用する **Install_Path** は、**CA Virtual Assurance** または **CA Virtual Assurance** 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を示します。

デフォルト

- Windows x86 : C:¥Program Files¥CA
- Windows x64 : C:¥CA、C:¥Program Files (x86)¥CA、または C:¥Program Files¥CA
- UNIX、Linux : /opt/CA

用語

MIB オブジェクト、MIB 属性

これらの用語は同義語として使用されます。

第 2 章：機能および統合概要

このセクションには、以下のトピックが含まれています。

[機能概要](#) (P. 21)

[プラグインと統合](#) (P. 33)

機能概要

SystemEDGE は、軽量で、展開することが簡単なパッケージで、以下のシステム管理機能を提供します。

- すべての SNMP 管理者によって使用される SNMP メトリックを収集します。
- メトリック値、プロセスおよびサービス、プロセス グループ、ログ ファイル、ディレクトリ、および Windows イベントのようなオブジェクトをモニタします。これらのオブジェクトは、SNMP トラップによって通知をトリガする設定可能な MIB テーブルを使用してモニタされます。
- セルフ モニタ テーブルおよびプロセス モニタ テーブルを使用して、オブジェクト モデルの状態を管理します。これにより、モニタされたオブジェクトの重大度の集約、オブジェクト ステータスの更新、および状態変更トラップの送信が容易になります。
- 履歴テーブル、および基準値作成および傾向分析を行うためのキューブ ファイル内のメトリックの収集。
- ファイル ベースのリモート エージェント設定およびリモート展開 (CA Virtual Assurance または CA Server Automation に統合された場合)。
- レジストリ エントリのような、モニタリング用のカスタム MIB オブジェクトを追加できる拡張可能な MIB。
- モニタ エントリが指定されたしきい値を超過するときにアクションをトリガする機能。
- 追加のモニタリング機能用のプラグイン モジュールである Application Insight Module (AIM) を許可する柔軟なアーキテクチャ。

関連項目

[コンセプト](#) (P. 41)

モニタリング機能

しきい値または他の例外用の特定の管理情報ベース（MIB）オブジェクトをモニタするために **SystemEDGE** エージェントを設定できます。数百台のシステムが存在する大企業を管理する場合は、エージェントによってモニタされる情報を設定できます。エージェントによってモニタされる情報を設定することにより、管理オーバーヘッドを増加させることなく、大規模な実装を保守するために必要な例外による管理が実現できます。

注: 各エージェントの収集およびモニタリング機能の詳細については、「概念」の章を参照してください。

サポートされるモニタの最大精度は 64 ビットです。SystemEDGE エージェントは以下のタイプのモニタリングを提供します。

セルフ モニタリング

エージェントがサポートする任意の整数ベース MIB オブジェクトのモニタリングを提供します。セルフ モニタ テーブルにエントリを作成して、モニタするオブジェクト、比較演算子、しきい値、および重大度を指定できます。エージェントは作成されたエントリに基づいてオブジェクトを自動的にモニタします。エージェントは特定のポーリング間隔でオブジェクトをモニタし、指定されたしきい値および重大度値に従って現在の状態を維持します。状態変更しきい値の超過が発生すると、エージェントは状態変更トラップを送信します。

注: セルフ モニタリングの詳細については、「セルフ モニタリング」の章を参照してください。

プロセスとサービスのモニタリング

任意の単一のプロセス インスタンスまたは Windows サービスのモニタリング機能を提供します。プロセス モニタ テーブルにエントリを作成して、プロセスまたはサービスが実行されているかどうかをモニタしたり、指定されたしきい値に対してプロセス テーブル MIB オブジェクトをモニタしたりできます。エージェントはプロセスをモニタし、指定されたしきい値および重大度値に従って現在の状態を維持します。状態変更しきい値の超過が発生するか、プロセスの状態（実行中または停止中）が変更されると、エージェントは状態変更トラップを送信します。

注: プロセスおよびサービスに対するモニタリングの詳細については、「プロセスとサービスのモニタリング」の章を参照してください。プロセス モニタは単一のプロセス インスタンスしかモニタできないので、複数のプロセス インスタンスをモニタするには、プロセス グループ モニタを使用します。

プロセス グループ モニタリング

プロセスのグループを定義し、そのグループをモニタする機能を提供します。プロセス グループ モニタ テーブルに、プロセス グループを定義するエントリを作成できます。エージェントはグループをモニタし、プロセス グループが変更されると、トラップを送信します。エージェント モニタは、グループ内のすべてのプロセスによって消費されるメモリまたは CPU 時間などのプロセス グループ メトリックを提供します。必要な場合、セルフ モニタがプロセス グループ メトリックをモニタできます。

注: プロセス グループ モニタリングの詳細については、「プロセス グループ モニタリング」の章を参照してください。

ログ ファイルおよびディレクトリ モニタリング

正規表現として指定された文字列を検索して、任意の UTF-8 エンコーディング システムまたはアプリケーション ログ ファイルのモニタリングを提供します。ログ モニタ テーブルにエントリを作成して、エージェントが、指定されたログ ファイルでユーザ定義の正規表現と一致する行をモニタし、一致が発生すると、トラップを送信することができます。送信されたトラップに含まれる重大度をモニタに関連付けることができます。ログ モニタ テーブルにエントリを適宜設定すると、ディレクトリ内のファイルのサイズおよび数もモニタできます。最大 **2 GB** のサイズのログ ファイルをモニタできます。

注: ログ ファイルおよびディレクトリのモニタリングの詳細については、「ログ ファイル モニタ」の章を参照してください。

Windows イベント モニタリング

イベント ソースなどのさまざまなフィルタを使用して、**Windows** イベント ログ エントリのモニタリングを提供します。**NT** イベント モニタ テーブルにエントリを作成して、エージェントが、指定されたイベント ログでユーザ定義の正規表現と一致するイベントをモニタし、一致が発生すると、トラップを送信するようにできます。

注: Windows イベント モニタリングの詳細については、「Windows イベント モニタリング」の章を参照してください。

履歴の収集

マネージャ側ベースライニングおよび傾向分析用の履歴データ収集を提供します。履歴制御テーブルにエントリを作成して、エージェントが時間経過に伴ってメトリックを収集するようにできます。収集されたメトリックは、特定の期間中の平均システム パフォーマンスの図を表示するために使用されます。

注: 履歴収集の詳細については、「履歴の収集」の章を参照してください。

64 ビット メトリックのスケールのモニタ

モニタのしきい値を指定して、64 ビット メトリックをモニタできます。たとえば、64 ビットのしきい値を指定すると、9,000,000,000,000,000,001 バイト (9 の * 10^{18} に 1 を加えた数) は有効な 64 ビット値です。ただし、 10^{18} のスケールファクタおよび 9 のしきい値を定義することにより、9 EB (エクサバイト) のしきい値を定義できます。スケールファクタは、モニタの精度を定義します。1 パーセントの精度のために、 10^{16} のスケールファクタを定義できます。対応するしきい値は 900 です。

64 ビット (2^{64}) メトリックに対して、モニタのスケールファクタが 2^{32} である場合、最大のモニタ値は 2^{32} です。

プロセス モニタリングの拡張機能

SystemEDGE は、プロセスの CPU 使用率を新しい変数 (processTable 内の processTimePermil) の割合として提供します。プロセス モニタリング機能およびプロセス グループ モニタリング機能が、processTimePermil を使用して提供されています。

processTimePermil は、前回のサンプル間隔でのプロセスの CPU 使用率を千分率で表した値です。変数値はシステムによって異なります。値は、CPU ティック数を総 CPU ティック数で除算した値の差異です。SNMP では小数がサポートされていないので、報告される値は 1000 倍されています。たとえば、CPU の千分率 505.833 が 505833 として返されます。パーセントに変換するには、返された値を 10000 で除算します。

オブジェクト集計

システム管理 MIB に新しい集計テーブルが追加され、異なる重大度の複数のモニタによってモニタリングされているオブジェクトが集計できるようになりました。集計テーブルは、セルフ モニタ テーブルとプロセス モニタ テーブルの新しいクラス、インスタンス、および属性のプロパティを使用して、モニタが同じオブジェクトを参照しているかどうかを確認します。全体的なオブジェクト状態は、しきい値を超えたモニタの中で最大の重大度を持つモニタの状態変更トラップを送信することによって確認されます。

多層オブジェクト インスタンス階層

オブジェクト インスタンスをそのホストされる階層に関連付けるために、`//`と`/`とが、`//hierarchy/instance` のように区切り文字として使用されます。階層は、オブジェクト インスタンスが 2 層より多い層でサポートされるような多層となることができます。

階層は、ホスト名 `//host/instance` の形式および

`//<any1>/<any2>/.../<anyN>/<instance>` の形式とすることができます。ローカル システムのオブジェクト インスタンスの場合、インスタンスのみが指定でき、先頭に `//` が自動的に付加されます。仮想オブジェクト インスタンスを示すために、先頭に「`lpar:`」のようなネームスペースを付加できます。たとえば、LPAR 上のオブジェクト インスタンスは、`lpar://hmc1/hyper1/lpar1/disk1` として表されます。

オブジェクト インスタンスのスーパー集計は、すべての層ではなく、最下位の 2 層のみに対して提供されます。たとえば、`//host/*` は、ホスト上のすべてのインスタンスを集計します。すべてのオブジェクト インスタンスを集計する、特別なスーパー集計 `//*` は、スーパー集計の選択したレベルに応じて作成されます。

状態管理

SystemEDGE エージェントは、作成されたセルフ モニタとプロセス モニタから取得されるステータス情報を計算します。モニタには、それぞれにしきい値条件と重大度が割り当てられます（モニタ 1: 「70% を超えると警告」、モニタ 2: 「90% を超えると重大」）。いくつかのモニタは、すべての関連するモニタの最悪の状態を持つ、単一の `MonitorObject` にまとめられます。エージェントは、状態変更トラップを通してステータス情報を送信します。CA Virtual Assurance などのステートフルマネージャはこのステータス情報にアクセスして解釈し、企業内のシステム、プロセス、サービス、およびアプリケーションについて、状態を中心とした情報表示を行います。

注: SystemEDGE 状態管理モデルの詳細については、「コンセプト」の章を参照してください。

サポートされている MIB

MIB は、エージェントによって制御される要素の情報を格納するために使用される情報ベースとして機能します。MIB は、それぞれが管理情報のアイテムであるオブジェクトの構造化された収集です。

管理システムは、管理対象リソースを、その MIB オブジェクトの値を読み取ることによってモニタします。また、SNMP SET 要求を通してリソース MIB 内のオブジェクトの値を変更（設定）することによって、リソースを制御します。

MIB は、特定のリソースに関連する管理オブジェクトを既定する MIB 規格で定義されます。MIB 規格はまた、オブジェクト収集が構造化される方法も定義します。MIB モジュールは、管理システムおよびエージェントの両方で使用されるデータ定義ドキュメントに似ています。

SystemEDGE エージェントは、次の MIB からのメトリックの収集をサポートします。

- CA Systems Management Empire MIB
- MIB-II (RFC 1213)
- Host Resources MIB (RFC 1514)
- IP-MIB (部分) (RFC 4293)
- IF-MIB (部分) (RFC 2233)
- TCP-MIB (部分) (RFC 4022)
- UDP-MIB (部分) (RFC 4113)

エージェントは、SNMP トラップ採用による柔軟なモニタ概念を使用して、収集した情報をこれらの MIB から、SNMP データを受け付ける任意の管理システムに送信できます。

CA Systems Management Empire MIB

Systems Management Empire MIB はプライベートな企業 MIB で、基になるシステムおよびアプリケーションのヘルス状態とパフォーマンスをモニタリングするためのオブジェクトを提供しています。この MIB は、次のオブジェクトの管理情報を定義します。

CA Systems Management Empire MIB

- システムの詳細
- マウントされたデバイス
- カーネルおよびシステムのパラメータ
- 起動設定
- ネットワーク、ストリーム、および I/O バッファの統計情報
- NFS (Network File System) および RPC (Remote Procedure Call) の統計情報
- コンテキスト スイッチ数およびページ フォールト数などのカーネル パフォーマンスの統計情報
- ファイル システム
- ユーザおよびグループ
- プロセス
- プロセス間通信
- システム リソース
- 論理パーティション
- ディスクおよび CPU の統計情報
- レジストリおよび Windows イベント ログなど Windows 固有の統計情報

Systems Management Empire MIB には、エージェントの自律モニタリングおよびデータ ストレージ機能を設定するために、以下のテーブルが用意されています。

- セルフ モニタ テーブル
- プロセス モニタ テーブル
- プロセス グループ モニタ テーブル
- ログ モニタ テーブル
- NT イベント モニタ テーブル
- 履歴制御テーブル

Systems Management Empire MIB にはまた、定義済みモニタおよびプロセス モニタから取得された、モニタ対象オブジェクトの状態に関する情報を提供するテーブルも用意されています。

SystemEDGE エージェントが Systems Management Empire MIB を使用方法の詳細については、「コンセプト」および「Systems Management Empire MIB」の章を参照してください。

注: Systems Management Empire MIB オブジェクトのサポートは、プラットフォームによって異なります。各プラットフォームでサポートされていない Systems Management Empire MIB オブジェクトのリストについては、「SystemEDGE リリースノート」を参照してください。

MIB-II

MIB-II は、ネットワーク インターフェースおよびプロトコル統計についての情報を提供する標準 (RFC 1213) です。この MIB には、以下のグループに関する情報が含まれます。

- システム
- インターフェース
- アドレス変換
- インターネットプロトコル (IP : Internet Protocol)

注: SystemEDGE では、IF-MIB ファイルの ifXTable のみサポートされます。

- Internet Control Message Protocol (ICMP)
- Transfer Control Protocol (TCP)
- User Datagram Protocol (UDP)
- 簡易ネットワーク管理プロトコル (SNMP : Simple Network Management Protocol)

注: Windows のリリースによっては、ワイヤレス ゼロ構成サービスが有効な場合、SystemEDGE がシステムから ifXTable データを取得できない場合があります。このサービスは、排他モードで NDISUIO ドライバを使用するため、SystemEDGE が NDISUIO ドライバにアクセスできない場合があります。これは、Window 2008、Windows Vista、Windows 7 上では発生しません。

それ以前の Windows リリースで ifXTable データが必要な場合、ワイヤレス ゼロ構成サービス、または ndisuio.sys を使用するすべてのサービスを無効にする必要があります。

MIB-II オブジェクトのサポートは、プラットフォームによって異なります。各プラットフォームでサポートされていない MIB-II オブジェクトのリストについては、SystemEDGE リリース ノートを参照してください。

Host Resources MIB

Host Resources MIB は、一般的なホストシステムに管理情報を提供するために Internet Engineering Task Force (IETF) によって定義されています。MIB には、以下のようなアセット管理に特に役立つ情報が含まれています。

- ファイルシステムやディスクパーティションなどのストレージ領域
- 実行中およびインストール済みのソフトウェア
- キーボード、ディスク、ネットワークカードなどのシステムデバイス

Host Resources MIB と共に SystemEDGE を使用すると、アセットの追跡を自動化し、インストールされたハードウェアとソフトウェアの現在の図を表示できます。SystemEDGE は、システムが正しく設定されているかどうかと、現在のパッチおよびサービスパックが含まれているかどうかを判断できます。この情報は、システム管理の簡素化、パフォーマンスの向上、およびセキュリティリスクの軽減に役立ちます。

SystemEDGE エージェントが Host Resources MIB を使用方法の詳細については、付録「Host Resources MIB」を参照してください。

注: Host Resources MIB オブジェクトのサポートはプラットフォームによって異なります。各プラットフォームでサポートされない Host Resources MIB オブジェクトのリストについては、SystemEDGE リリースノートを参照してください。

IP-MIB テーブル

SystemEDGE では以下の IP MIB テーブルがサポートされます。

MIB テーブル	Windows	Solaris	HP-UX	AIX	Linux
ipSystemStatsTable (RFC 4293)	No	Yes	No	Yes	Yes
ipIfStatsTable (RFC 4293)	No	No	Yes	No	No
ipAddressPrefixTable (RFC 4293)	No	No	Yes	No	Yes
ipNetToPhysicalTable (RFC 4293)	No	Yes	Yes	No	No
ipAddressTable (RFC 4293)	No	Yes	Yes	Yes	Yes
ipDefaultRouterTable (RFC 4293)	No	Yes	Yes	Yes	Yes
icmpStatsTable (RFC 4293)	Yes	Yes	No	Yes	Yes
tcpConnectionTable (RFC 4022)	Yes	Yes	Yes	Yes	Yes
udpEndpointTable (RFC 4113)	Yes	Yes	Yes	Yes	Yes

注:サポートされる IP MIB オブジェクトはプラットフォームによって異なります。各プラットフォームでサポートされていない IP MIB オブジェクトのリストについては、SystemEDGE リリース ノートを参照してください。

MIB 拡張

Systems Management Empire MIB は、ユーザが拡張できます。MIB に拡張を追加すると、ベース MIB の対象とならない、企業にとって重要なオブジェクトを管理できます。Systems Management Empire MIB は、以下の方法でカスタマイズできます。

- カスタマイズされた管理用のスカラ MIB オブジェクトを作成します。カスタマイズされた各 MIB オブジェクト番号および MIB オブジェクトタイプと、新しい MIB 変数がクエリ実行または設定されたときに実行されるスクリプトまたはプログラムの名前を使用して、SystemEDGE を設定できます。
- カスタム MIB オブジェクトを作成して、Windows レジストリとパフォーマンス データを管理します。外部プログラムまたはスクリプトを使用しなくても追加のレジストリ パラメータおよびパフォーマンス データを報告するように SystemEDGE を設定できます。この機能を使用すると、パフォーマンス レジストリをとおしてパフォーマンス データを利用できるようにするアプリケーションのヘルスをモニタできます。

注:カスタム MIB オブジェクトの作成の詳細については、「カスタム MIB オブジェクト」の章を参照してください。Windows レジストリとパフォーマンス データを管理するオブジェクトの作成の詳細については、「Windows レジストリとパフォーマンスの MIB オブジェクト」を参照してください。

修正処置

SystemEDGE エージェントは、アクションを呼び出すことで、システムの危機的な状況に自動的に対応できます。アクションは、一定の条件を満たしたときに実行される特定のコマンドまたはスクリプトです。たとえば、失敗したプロセスをエージェントが再起動できるようにしたり、システムへの不正なアクセスがあった場合にエージェントが管理者に電子メールを送信できるようにするアクションを指定できます。

アクションを使用するときは、コマンドのフルパス（パラメータがある場合はパラメータ付きで）を指定する必要があります。環境変数の使用がサポートされます。エージェントは、モニターテーブルエントリの条件を満たすたびにこのコマンドを実行します。

アクションの指定および詳細な例の詳細については、以下の章を参照してください。

- 「セルフ モニタリング」
- 「プロセスとサービスのモニタリング」
- 「プロセス グループ モニタリング」
- 「ログ ファイル モニタリング」
- 「Windows イベント モニタリング」

プラグインと統合

SystemEDGE エージェントの SNMP アーキテクチャを使用すると、任意の SNMP ベースのマネージャと統合できます。エージェントは、特定の管理製品がなくても設定でき、SNMP 情報を収集するあらゆるアプリケーションにトラップを送信できます。多くの CA 製品は、エージェントとの強化された統合を特長としているため、そのすべての機能を活用でき、そのデータを解釈してエージェントをリモートで設定および展開するためのインターフェースを利用できます。

具体的な管理基準として、CA では Application Insight Module (AIM) と呼ばれる SystemEDGE プラグインモジュールを用意しています。AIM は、エージェントの既存のアーキテクチャと統合され、特定のアプリケーションまたはサービスをモニタするツールを提供することでその機能を拡張します。

このセクションでは、SystemEDGE との統合と、エージェント プラグインとして使用できる AIM を特長とする CA 製品について説明します。

Application Insight Module (AIM)

SystemEDGE エージェントには、初期化時にオプションの **Application Insight Module (AIM)** をロードできるプラグインアーキテクチャが備わっています。これらの AIM は、アプリケーション固有のセマンティックナレッジをサポートするための拡張可能で柔軟な手法を提供します。

以下の AIM は、SystemEDGE 状態管理モデルに準拠し、マネージャ ユーザ インターフェースを通して展開および設定をサポートします。

VMware vCenter Server AIM

VMware vCenter Server の管理下にあるシステムを管理する機能を提供します。

リモート モニタリング AIM

リモート（エージェントが存在しない）システムの管理とモニタリングの機能を提供します。

サービス レスポンス モニタ (SRM)

管理対象ノード上のサービス レスポンスをモニタします。サービス レスポンス モニタ (SRM) 用 AIM は、SystemEDGE 状態管理モデルを利用するモニタを作成し、**CA Virtual Assurance** ユーザ インターフェースを通して展開および設定をサポートします。

Cisco Unified Computing System (UCS)

Cisco UCS Manager からデバイスと統計のデータを収集し、外部アプリケーションに対して **SNMP** サポートを提供します。

LPAR AIM (AIX PowerVM)

LPAR を含むシステム全体を管理およびモニタする機能を提供します。

Microsoft Cluster Server AIM

Microsoft クラスタをリモートシステムからモニタします。MSCS モニタリングでは、CPU とメモリの使用率などのメトリックを収集するためのクラスタおよび個々のクラスタ ノードへのリモートアクセスが必要です。

Microsoft Hyper-V AIM

Hyper-V Server を管理およびモニタする機能を提供します。

Exchange Server および Active Directory 用 AIM

クラウドおよび社内運用の両方のインフラストラクチャ上の **Exchange Server** および **Active Directory** をモニタする機能を提供します。**CA Server Automation** には適用できません。

Solaris ゾーン AIM

コンテナとゾーンを実行するように設定された **Solaris 10** システムを管理およびモニタする機能を提供します。

KVM AIM

リモートシステムからカーネルベースの仮想マシン環境をモニタします。KVM AIM は 1 つ以上の RHEV マネージャと通信します。CA Server Automation には適用できません。

Citrix XenServer AIM

Citrix XenServer 環境をモニタします。AIM はプールマスタのみと通信します。

CA Virtual Assurance

CA Virtual Assurance は、仮想リソースを自動的にモニタ、再設定、プロビジョニングしてサービス指向の複雑なデータセンターの負荷需要を動的に満たす、ポリシーベースの製品です。CA Virtual Assurance は、サービス指向アーキテクチャ (SOA) に基づいて構築されており、サーバを最適にプロビジョニングして必要なタスクを実行できるように、データセンターを継続的に分析します。

CA Virtual Assurance は、広範なシステムをモニタおよび管理するための基盤として SystemEDGE を使用します。

機能によっては、CA Virtual Assurance ユーザインターフェースと共にエージェントを使用するときのみ使用できます。これらの製品と共に SystemEDGE を使用すると、以下の独自の機能を利用できます。

- プラットフォーム管理モジュールをとおした、エージェントの状態管理データの視覚化。すべてのモニタの状態を表示し、状態を伝達している重大なモニタを検索および診断できます。
- リモートシステムへのエージェント展開。この機能には、SNMP コミュニティや他のインストール設定などの設定を行い、それらの設定を持つエージェントを任意の管理システムに展開する機能が含まれます。
- リモートシステムのエージェント設定。CA Virtual Assurance は、ファイルベースの設定ソリューションを特長としているため、モニタエントリを作成し、SystemEDGE エージェントの任意の側面を設定し、任意の数の管理システムにその設定を展開することができます。

- エージェントのウォーム スタート。リモートシステムにエージェントを展開するとき、ほとんどの場合エージェントの完全な再起動が必要ありません。エージェントはウォーム スタートを実行し、実行時に設定変更を実装します。
- 管理対象エージェントへの **SNMP** ベースの変更を通知する機能や、**CA Virtual Assurance** によって開始されなかった変更をすべて上書きするオプションなど、設定ロック ダウン オプション。
- サービス レスポンス モニタ **AIM** の展開および設定。
- 仮想環境モニタリング **AIM** プラグインを幅広く選択可能な仮想プラットフォーム管理。

CA Virtual Assurance は、**SystemEDGE** エージェントの機能を強化します。また、**SystemEDGE** を使用して環境内の物理システムおよび仮想システムをすべて管理する場合に推奨される製品です。

CA Virtual Assurance の詳細については、**CA Virtual Assurance** のドキュメントを参照してください。

CA Server Automation

CA Server Automation は、物理リソースおよび仮想リソースを自動的にモニタ、再設定、プロビジョニングしてサービス指向の複雑なデータ センターの負荷需要を動的に満たす、ポリシーベースの製品です。CA Server Automation は Service Oriented Architecture (SOA) 上に構築されており、データ センターを継続的に分析して、必要なタスクを実行できるようにサーバが最適な方法でプロビジョニングされるようにします。

CA Server Automation には、SystemEDGE エージェントなど、埋め込みの CA Virtual Assurance 機能が含まれています。CA Server Automation 内で SystemEDGE を使用すると、データ センターのシステム中心の管理ビューを表示できます。SystemEDGE によって示される情報があると、データ センター内の物理システムおよび仮想システムの自動化について、より多くの情報に基づいて決定を下すことができます。CA Virtual Assurance 機能とエージェントを組み合わせると、CA Virtual Assurance AIM を介して仮想システムと物理システムを管理できます。

CA Virtual Assurance プラットフォーム管理モジュールは、CA Server Automation ユーザ インターフェイスに表示する情報を SystemEDGE エージェントから収集します。CA Server Automation ユーザ インターフェイスには、すべての物理システムおよび仮想システムのトポロジ ビューが、複数の異なるコンテキストで表示されます。このビューで SystemEDGE によって管理されているシステムを選択すると、モニタされているオブジェクト状態サマリ、定義済みモニタ、およびさまざまなシステム メトリックが表示されます。ユーザ インターフェイスには、また仮想モニタリング AIM によって有効によるシステム アクションを実行するオプションもあります。CA Server Automation の内の CA Virtual Assurance 機能の全スライトに対するアクセス権があり、CA Server Automation イベント ビューで SystemEDGE トラップを表示できます。

CA Spectrum

CA Spectrum は、SystemEDGE エージェントによる検出されたシステムの管理をサポートします。CA Spectrum で SystemEDGE を管理すると、分散ネットワークのコンテキスト内でシステム管理を行うことができます。エージェントは、パフォーマンスの低いシステムや、ネットワークの動作に影響を与える可能性がある停止したプロセスまたはアプリケーションに関して、CA Spectrum にアラートを送信できます。

SystemEDGE エージェントによって収集された SNMP メトリックを表示でき、Spectrum OneClick インターフェースからセルフ モニタリングテーブルのエントリを表示できます。CA Spectrum は、エージェントによって送信されたトラップを収集して、それらをイベントとして表示できるため、OneClick からの例外を管理できます。特定の SystemEDGE トラップが収集されたときにアラームを生成するルールを使用して CA Spectrum アラーム ビューを設定できるため、ネットワーク障害の場合と同じ方法で特定のシステムの問題を検出して解決できます。

CA Spectrum 9.2 では、状態管理機能を有効にする SystemEDGE r5.0 Systems Management Empire MIB がサポートされます。SystemEDGE 5.0 によって提供される埋め込みの OneClick テーブルを使用して、モニタを追加、編集、および削除できます。CA Spectrum r9.2 は、SystemEDGE エージェントのリリース（最新または以前の）を動的にチェックし、リリース固有の情報を表示します。

CA Spectrum では SystemEDGE AIM もサポートされます。CA Spectrum 9.2 で AIM トラップを受信すると、トラップにはエージェント タイプ、エージェント名、およびアラーム タイプが含まれています。

注: OneClick で SystemEDGE エージェントを使用すると、CA Virtual Assurance に固有のいくつかの機能は利用できません。これらの機能の詳細については、「コンセプト」の章を参照してください。

エージェントの全機能を有効にするには、CA Virtual Assurance でエージェントを管理する必要があります。詳細については、CA Virtual Assurance のドキュメントを参照してください。

CA eHealth

`nhAddSysEdgeMonEntries` コマンドを実行すると、重大な CA eHealth プロセスとシステム ログをモニタするように SystemEDGE エージェントを設定できます。このコマンドは、`sysedge.cf` ファイルにエントリを追加し、SystemEDGE エージェントを停止して再起動することで変更を実装します。

注: `nhAddSysEdgeMonEntries` コマンドの詳細については、「CA eHealth Administration Reference」を参照してください。

CA eHealth と共に SystemEDGE を使用すると、CA eHealth ポーラーは SystemEDGE エージェントからデータを収集して CA eHealth データベースに格納するため、CA eHealth レポートおよびリアルタイム モニタリング ツールがデータを利用できます。システムで At-a-Glance (AAG)、Trend、Top N、What-If Capacity Trend、System Health、および MyHealth の各レポートを実行して、キャパシティ計画の実行、サービスの問題の正確な文書化、および潜在的な問題のトラブルシュートを行うことができます。

注: CA eHealth アプリケーションの詳細については、「CA eHealth System and Application Administration Guide」を参照してください。

CA eHealth では、レガシー モニタ（つまり、SystemEDGE r5.0 の機能を使用しないモニタ）を使用した SystemEDGE エージェントの実行がサポートされます。CA eHealth および eHealth AdvantEDGE View と共にエージェントを使用すると、CA Virtual Assurance に固有のいくつかの機能を使用できません。レガシー モニタおよびレガシー モード（CA Virtual Assurance を使用しないでエージェントを実行）の詳細については、「コンセプト」の章を参照してください。エージェントの全機能性を有効にするには、CA Virtual Assurance でエージェントを管理する必要があります。詳細については、CA Virtual Assurance のドキュメントを参照してください。

CA NSM

CA NSM は、SystemEDGE エージェントと統合できるため、エージェントデータを使用して、検出されたシステムを管理できます。SystemEDGE エージェント管理機能と CA NSM DSM の機能を組み合わせると、各エージェントにより固有の管理対象オブジェクトがモニタされる、階層状のモニタリングアーキテクチャを利用できます。

CA NSM には SystemEDGE エージェントの DSM のクラス およびポリシーが含まれており、エージェントをポーリングおよび検出してシステム情報を取得できます。DSM のモニタ対象オブジェクトは、オブジェクトテーブルエントリが表示されるたびに作成され、CA NSM はオブジェクトテーブルから送信されたトラップを処理できます。SystemEDGE エージェントビューでは、エージェントのサマリを表示し、CA NSM の内のセルフ モニタリングテーブルおよびプロセス モニタテーブルにエントリを作成できます。

CA NSM r11.2 から、CA NSM は SystemEDGE エージェントおよび SRM AIM の SystemEDGE 状態管理モデルにより作成および維持される状態を解釈できるようになりました。CA NSM と CA Virtual Assurance を統合すると、Management Command Center のコンテキスト内リンクから CA Virtual Assurance ユーザーインターフェースにアクセスして、リモート エージェント展開および設定機能にアクセスできます。

注: CA NSM と CA Virtual Assurance の統合の詳細については、*CA Virtual Assurance* のドキュメントを参照してください。

第 3 章: コンセプト

このセクションには、以下のトピックが含まれています。

[アーキテクチャの概要](#) (P. 42)

[モニタリング アーキテクチャ](#) (P. 50)

[状態管理モデル](#) (P. 70)

[CA Virtual Assurance との統合](#) (P. 81)

[AIM アーキテクチャ](#) (P. 85)

[SystemEDGE エージェントの使用のガイドライン](#) (P. 87)

アーキテクチャの概要

SystemEDGE エージェントは、システムをモニタするために軽量で柔軟な SNMP アーキテクチャを使用します。SNMP は、TCP/IP ベースのネットワークおよびデバイスを管理するための標準です。その最も基本的なレベルで、SystemEDGE は、SNMP フレームワークにおけるエージェントの役割を果たし、管理対象オブジェクトに対して SNMP ベースの命令を実行します。命令は、管理システムからのものである場合とエージェント自体からのものである場合があります。また、これらの命令は、オブジェクトに関する情報に対する要求である場合と、システムパラメータを変更するための要求である場合があります。SystemEDGE エージェントはこれらの要求を実行し、要求しているアプリケーションに応答を送信します。エージェントは、標準的な SNMP プロトコルに準拠して情報を交換します。その結果、SNMP によって通信できるどんなマネージャもエージェントと統合できます。

SystemEDGE エージェントは MIB 属性によってシステム メトリックにアクセスします。MIB は、共通の属性形式に準拠した管理情報のストアです。エージェントは、サポートおよび管理するすべての MIB 内の属性を収集したり変更したりすることができます。

注: サポートされる MIB の詳細については、「はじめに」を参照してください。

SystemEDGE エージェントはデフォルトでは(インストール中にこのポートを変更できます) UDP ポート 161 上の SNMP によって通信します。SNMPv1 環境で、エージェントは、以下の標準の SNMP メッセージタイプを使用して管理情報を交換します。

GetRequest

MIB から特定のオブジェクトの値を取得します。管理システムは、システムに関する特定の情報を要求するとき、エージェントにこのメッセージを送信します。

GetNextRequest

次のオブジェクト インスタンスを取得します。

GetResponse

要求しているアプリケーションに要求された MIB オブジェクト情報を返します。

SetRequest

MIB オブジェクトの値を変更するようにエージェントに指定します。

Trap

管理システムに例外を通知します。**Trap** を使用すると、エージェントがマネージャに要請されないメッセージを送信できます。詳細については、[「SNMP トラップ」](#) (P. 45) を参照してください。

SNMPv2 環境で、管理情報を交換する標準の SNMP メッセージタイプは以下のとおりです。

GetBulkRequest

1 つ以上の **MIB** オブジェクトの値（表形式）を取得します。これは **GetNextRequest** に似ています。

この共通の SNMP ワークフロー（要求の取得または設定、検索、MIB 属性の変更、SNMP 応答）は、すべての SystemEDGE 機能の基本から構成されます。ただし、エージェントは、基本の SNMP コレクションを超えて、インテリジェントなモニタリングおよび状態管理に移動することができます。詳細については、[「モニタリングアーキテクチャ」](#) (P. 50) および [「状態管理モデル」](#) (P. 70) を参照してください。

SNMP のバージョンとアクセス

SystemEDGE は SNMPv1、SNMPv2c、および SNMPv3 プロトコルをサポートします。各プロトコルには一致の機能が含まれ、主な違いはメッセージのセキュリティです。エージェントは、プロトコルのいずれかのセキュリティ アクセス方法を使用して、SNMP メッセージを送信できます。

SNMPv1 と SNMPv2c のコミュニティ

SNMPv1 と SNMPv2c は、システム間でメッセージを送信するためにコミュニティを使用します。コミュニティは、エージェントと多数の管理システムの関係で、分散した通信のために認証とアクセス制御権限を定義します。コミュニティはコミュニティ名によって識別されます。コミュニティ名は、すべての SNMP メッセージのヘッダ部分に表示される、オクテットの文字列です。

エージェントは、SNMP メッセージヘッダのコミュニティ名を確認し、メッセージが認証されたものであるかどうかを判断してから、メッセージを処理します。コミュニティ名がエージェントによって受諾された名前と一致しない場合、エージェントは認証エラーを記録し、メッセージを削除します。

SystemEDGE エージェントは、デフォルトで定義されている **public** という名前の 1 つのコミュニティと共にのみインストールされます。このコミュニティは、任意のマネージャ システムの、エージェントの MIB オブジェクト値に対する読み取り専用アクセスを提供します。MIB 値を変更する前のインストール中またはインストール後に、読み取り/書き込みコミュニティ文字列を設定する必要があります。コミュニティ定義に IP アドレス リストを追加して、特定のシステムに対するアクセスを制限することができます。

注: SNMP コミュニティの設定の詳細については、「エージェントの設定」の章を参照してください。

SNMPv3 のユーザとセキュリティ

SNMPv3 は、ネットワーク上で認証と暗号化の組み合わせを使用して、安全なデバイス アクセスを提供します。

認証

パケットが送信中に変更されなかったことを確認するのに役立ちます。

暗号化

パッケージの内容を暗号化して、未承認のソースからの表示を防ぎます。

SNMPv3 ではセキュリティ モデルとセキュリティ レベルが使用されます。セキュリティ モデルは、ユーザおよびそのユーザが存在するグループのためにセットアップされた認証方法です。セキュリティ レベルは、セキュリティ モデル内のセキュリティの許可されたレベルです。セキュリティ モデルとセキュリティ レベルの組み合わせにより、SNMP パケットを処理するためのセキュリティ メカニズムが決まります。この組み合わせによって、SNMP デバイスの安全なデータ コレクションと機密情報の暗号化が有効になり、ネットワーク送信での露出を防ぐことができます。

SystemEDGE エージェントでは、SNMP ベースの通信に対して USM (User-based Security Model) が使用されます。指定された SNMPv3 ユーザ、認証と暗号化プロトコル、およびパスワードを使用するように、SystemEDGE エージェントを設定できます。SNMPv3 の有効化と設定の詳細については、「エージェントの設定」の章を参照してください。

SNMP トラップ

SystemEDGE は SNMPv1 トラップ、SNMPv2c トラップ、SNMPv3 トラップ、および通知を送信できます。設定されたしきい値がモニタされるオブジェクト変更の集計状態を超えたとき、または別の指定された条件が満たされたとき（ログ モニタの一致など）、エージェントは状態変更トラップを送信します。

SNMP トラップの設定は、SNMPv1 トラップと SNMPv2c トラップのトラップコミュニティ（トラップ先を含む）の設定、および SNMPv3 トラップの SNMPv3 ユーザの設定から構成されます。トラップ先は、エージェントが生成するトラップメッセージを受信する管理システムを示します。エージェントのモニタリング機能は、トラップコミュニティ内の管理システムにトラップメッセージを送信することによって、例外条件をレポートします。

注：トラップを送信するための SystemEDGE エージェントの設定の詳細については、「[エージェントの設定](#)」の章を参照してください。エージェントが送信できるトラップの種類の詳細については、「[モニタリングアーキテクチャ](#) (P. 50)」を参照してください。

SystemEDGE エージェントは、Systems Management Empire MIB の読み取り専用の trapCommunity テーブル内にトラップの設定情報を格納します。このアクセス情報を公開したくない場合に無効にする方法など、このテーブルの詳細については、「[エージェントの設定](#)」の章を参照してください。管理システムでのこのテーブルの表示の詳細については、*CA Virtual Assurance* および *CA eHealth* のドキュメントを参照してください。

SNMP テーブル

SystemEDGE エージェントは、テーブルを使用して、「マルチインスタンス」項目を表します。これらのテーブルのほとんどで、テーブルインデックスとして整数値が使用されます。インデックスは、テーブル内の項目を一意に識別するために重要で、エージェントの再起動中存続します。

たとえば、セルフモニタ テーブルでは整数インデックスが使用されます。以下のコマンド（sysedge.cf ファイル内）は、ディスク ドライブの領域使用率のために devTable 用のモニタを作成します。index(99) はこの特定のモニタ インスタンスと関連付けられます。

```
monitor devTableEntry 'C:' devCapacity 99 0x100 30 absolute >= 95 'exempl'
```

インデックス値 99 によって、SNMP クエリはモニタ エントリの現在値を取得できます。OID monCurrVal.99（記号の表記）は、以下のように SystemEDGE snmpget コマンドを使用してクエリできます。

```
snmpget -h <host> -c public -o 1.3.6.1.4.1.546.6.1.1.6.99
```

OID monCurrVal.99（記号の表記）は、記号の表記を使用してクエリすることもできます。

```
snmpget -h <host> -c public -o monCurrVal.99
```

インスタンスによるモニタの作成

表形式の MIB オブジェクト上でセルフ モニタを作成するときは、モニタされるインスタンスのインデックスを知る必要があります。いくつかの重要な MIB テーブルについては、エージェントはルックアップ機能（上記の例で使用）を提供します。これにより、以下の 2 つのオプションが有効になります。

- **sysedge.cf** 設定ファイルの使用

たとえば、**devTable** には整数インデックスがありますが、ルックアップ機能を使用すると、ドライブ文字を使用して定義される **devTableEntry** をモニタできます。**SystemEDGE** は、ドライブ文字の一致がないか **devTable** エントリを検索します。**devTableEntry** にインデックス エントリ 3 がある場合、モニタのオブジェクト識別子はこのインデックス (**devCapacity.3**) に設定されます。

- **SNMP** の使用

モニタの MIB 属性内 (**monObjClass**、**monObjInstance**、**monObjAttribute**) でテーブル名、インスタンス名、および属性名を指定すると、**SystemEDGE** はモニタのオブジェクト識別子を決定します。

以下の例では、指定された属性を使用して、モニタされる OID の正しい **devTable** インデックスを自動的に取得する、インデックスが **111** のモニタ エントリを作成します。ユーザが行ステータスを設定すると、モニタが実際に作成されます。

```
set monObjClass.111 = "devTableEntry"
set monObjInstance.111 = "C:"
set monObjAttribute.111 = "devCapacity"
set monRowStatus.111      = active(1)
```

注: **SystemEDGE** エージェントに用意されている **snmpset** コマンドを使用して、**SNMP** 値を設定することができます。詳細については、「**コマンドラインユーティリティ**」の章を参照してください。

整数値を使用してインデックス化されないエージェント テーブルは、**mirrorMonitorTable** と **the mirrorAggregateTable** の 2 つだけです。これらのテーブルでは、特定の管理対象オブジェクトのアクセスを許可するインデックスとして、オブジェクトクラス、インスタンス、および属性値が、使用されます。

設定ファイル

SystemEDGE エージェントの操作は、以下の設定テキスト ファイルの内容に基づきます。

- sysedge.cf
- sysedgeV3.cf

エージェントのインストールにより、システム情報、SNMP コミュニティ、CA Virtual Assurance マネージャ登録、データ ディレクトリ（このドキュメント内では CASYSEGE_DATA と呼ばれる）など、インストール中に定義される基本情報を含む sysedge.cf ファイルが、CASYSEGE¥config 内に作成されます。

開始した SystemEDGE エージェントは、インストール中に作成されたファイルを読み取って、それがどのように動作するかを決定し、CASYSEGE_DATA のサブディレクトリ内に sysedge.cf ファイルのコピーを作成します。

インストール後のランタイム設定変更を行うには、このバージョンのファイルを使用します。インストール中に定義された設定を編集し、他の設定を定義することができます。

注: インストール ディレクトリとデータ ディレクトリの場所の詳細については、「エージェントの設定」の章を参照してください。このドキュメントでは、インストール ディレクトリを CASYSEGE と呼びます。

sysedgeV3.cf ファイルには SNMPv2c および SNMPv3 の設定情報が含まれます。インストール中に SNMPv2c および SNMPv3 を設定することはできませんが、（エージェントの初回開始時にデータ ディレクトリのサブディレクトリにコピーされた）sysedgeV3.cf ファイルをデータ ディレクトリに変更して、SNMPv3 の操作を有効にし、SNMPv3 ユーザ、アクセス リスト、およびセキュリティ プロトコルを定義することができます。sysedgeV3.cf を中央の設定リポジトリに保持するか、（このファイルの自動展開を利用するために）CA Virtual Assurance のポリシーベースの設定機能を使用することをお勧めします。

注: `sysedge.cf` および `sysedgeV3.cf` を使用した基本的なプロパティの設定、およびエージェントの `SNMPv3` の設定の詳細については、「[エージェントの設定](#)」の章を参照してください。

`sysedge.cf` ファイルはまた、エージェントのモニタリング設定を制御します。`SNMP Set` 要求を送信する以外のモニタリング エントリの設定方法（`CA Virtual Assurance` などのユーザ インターフェースまたは直接ファイル操作による）では、最終的にデータ ディレクトリ内の `sysedge.cf` ファイルにモニタリング情報が取り込まれます。エージェントは、このファイルを使用して、モニタリング操作をすべて実行するための命令を収集します。

注: エージェントのモニタリング機能の詳細については、「[モニタリング アーキテクチャ](#) (P. 50)」を参照してください。

`sysedge.mon` ファイルは、`sysedge.cf` ファイルに自動入力されない `SNMP SET` を使って行われた変更を保存するためのバックアップ ファイルとして機能します。また、このファイルには、モニタの現在の状態などといったランタイム情報も保存されます。`sysedge.cf` 内のエントリは `sysedge.mon` 内のエントリに優先します。エージェントが完全（コールド）再起動を実行するときは、エントリをモニタするためのデータ ディレクトリに含まれる `sysedge.mon` ファイルを最初に読み取ります。次に `sysedge.cf` ファイルを読み取り、既存のモニタを上書きして、その他のモニタを追加します。エージェントが部分的（ウォーム）起動を実行するときは、`sysedge.cf` ファイルのみを読み取ります。エージェントを設定するには、`CA Virtual Assurance` ユーザ インターフェース内での設定または直接操作を通じて、`sysedge.cf` ファイルを使用します。

モニタリング アーキテクチャ

SystemEDGE エージェントのモニタリング機能は、基本的な SNMP メトリック収集だけでなく、インテリジェント モニタリングおよび例外報告にまで及びます。エージェントがモニタする対象と、エージェントが管理システムにトラップを返すタイミングは、制御することができます。

エージェントは、そのモニタリングデータを Systems Management Empire MIB 内のモニタ テーブルに維持します。モニタ テーブル内にモニタ エントリを作成することは、エージェントに対して、テーブルを読み取り、エントリに対してアクションを起こす指示となります。エージェントは指示に従ってシステムを周期的にポーリングし、モニタ対象オブジェクトの戻り値が特定の条件（しきい値、プロセス状態、正規表現など）に一致した場合、トラップ先として定義されたすべてのシステムにトラップを送信します。

Systems Management Empire MIB には以下のモニタリング テーブルが含まれます。

セルフ モニタ テーブル (monitorTable)

重大度および結果のオブジェクト ステータスを制御する目的でモニタし、ユーザ指定のしきい値と比較するための MIB オブジェクトが格納されます。

プロセス モニタ テーブル (processMonTable)

ステータス（実行中であるかどうか）およびリソースの使用状況についてモニタするためのプロセスが格納されます。

プロセス グループ モニタ テーブル (processGroupMonTable)

ステータス（変更が発生したかどうか）についてモニタするためのプロセス グループが格納されます。

ログ モニタ テーブル (logMonitorTable)

指定された正規表現文字列についてモニタするためのログ ファイルおよびディレクトリが格納されます。

NT イベント モニタ テーブル (ntEventMonTable)

特定のイベントを検索するためのイベント ログが格納されます。

履歴制御テーブル (empireHistoryCtrlTable)

管理システムによって将来取得されるように、エージェントがモニタして履歴テーブルに保存する MIB オブジェクトのサンプル間隔とサンプル数が格納されます。

各テーブルには **MIB** 属性が含まれ、属性の値はエントリの作成時に指定します。テーブルにはインデックスが付いているので、各エントリは一意的な属性値を持ちます。エージェントがテーブルを読み取ると、読み取ったエントリ属性が、モニタするシステム属性および例外を報告するタイミングに関する指示になります。

すべてのモニタ テーブル エントリは、ローカルの **sysedge.cf** ファイルと **sysedge.mon** ファイルに維持されます。**sysedge.mon** ファイルは互換性の理由でのみ使用されます。ディレクティブを使用すると、**sysedge.cf** ファイルにエントリを直接入力できます。つまり、エージェントと連携する管理製品内でエントリを設定できます。エージェントを **CA Virtual Assurance** と組み合わせて使用するとき、モニタ エントリを作成し、それらのエントリを含むファイルを、1 台または複数の管理対象システム上のエージェントに展開できます。

SNMPv2 の行ステータス

SystemEDGE は、SNMPv2 で導入された行ステータスの概念を利用して、任意の MIB モニタ テーブルに含まれる行（エントリ）の作成、アクティブ化、および保守を管理します。**RowStatus** テキスト表記法は、行の作成および削除を管理します。また、概念上の行のステータス列に対する **SYNTAX** 句の値です。

RowStatus ::= TEXTUAL-CONVENTION

STATUS current

モニタ テーブル内に定義する各エントリは、テーブルの行になり、行ステータス値を持ちます。行ステータスによって、アクセス可能なデータを含むアクティブな行であるかどうかが決まります。行ステータスに指定できる値を以下に示します。

active

行は管理対象デバイスによって使用される状態にあることを示します。エージェントがモニタ エントリ内のモニタリング指示を実行するには、エントリに **active** の行ステータスが必要です。デフォルトでは、有効な情報を使ってエントリを作成すると、行ステータスは自動的に **active** に設定されます。

notInService

行はエージェント内に存在するものの、管理対象デバイスによって使用される状態にないことを示します。

notReady

行はエージェント内に存在するものの、管理対象デバイスによって使用されるために必要な情報がないことを示します。

createAndGo

管理ステーションによって指定されます。行を作成し、作成した行が管理対象デバイスによって使用されるようにします。

createAndWait

管理ステーションによって指定されます。概念上の行のインスタンスを作成するものの、作成したインスタンスが管理対象デバイスによって使用されないようにします。

destroy

管理ステーションによって指定されます。行を削除します。

notReady と destroy 以外のステータスは、行の作成中に入力できます。既存の行を削除するには、destroy ステータスを使用します。マネージャの Get 要求に応じて返されるのは、以下の 3 つの値のみです。

active

行は管理対象デバイスによって使用される状態にあることを示します。

notInService

エージェントに十分な情報があるにもかかわらず、行は管理対象デバイスによって使用される状態にないことを示します。

notReady

エージェントに十分な情報がないために、行は管理対象デバイスによって使用される状態にないことを示します。

注: モニタ エントリの行ステータス制御の詳細については、このガイドのモニタリングに関する章を参照してください。 RowStatus テキスト表記法の詳細については、RFC 1443 : Textual Conventions for SNMPv2 (英語) を参照してください。

ステートフル モニタリングの仕組み

ステートフル モニタリングでは、定義された重大度情報とオブジェクト モデルを使用して、モニタまたはモニタ セットによって表される各オブジェクトに状態を割り当てます。以下のテーブルがステートフル モニタリングをサポートしています。

- [セルフ モニタ](#) (P. 56)
- [プロセス モニタ](#) (P. 63)

セルフ モニタ テーブル内にエントリを作成するときは、特定のしきい値についてモニタするための MIB オブジェクト (OID) を、重大度およびオブジェクト情報と共に指定します。プロセス モニタ テーブル内にエントリを作成するときは、実行状態またはプロセス属性のしきい値についてモニタするためのプロセスを指定します。このプロセスによって、エージェントが重大度情報を使ってステートフル モニタを処理する方法が記述されます。エージェントはレガシー モニタを処理する場合とは異なる方法で動作します。レガシー モニタには重大度が含まれません。

注: セルフ モニタリングとプロセス モニタリングに関する具体的な情報については、「セルフ モニタリング」と「プロセス モニタリング」の章で各タイプのモニタリングに関するセクションを参照してください。

SystemEDGE エージェントがステートフル モニタリングの指示を遂行する仕組みを以下に示します。

1. プロセス グループ、ログ ファイル、または Windows イベント ログ内で、指定されたグループ変更、正規表現、またはイベントをモニタするため、ユーザが適切な Systems Management Empire MIB テーブル内にエントリを作成します。

CA Virtual Assurance を使用していない限り、sysedge.cf ファイルへの変更を認識させるには、エージェントを再起動する必要があります。エントリを作成し、CA Virtual Assurance を通じて変更を展開すると、エージェントはウォーム スタートを実行し、起動中にエントリの読み取りを開始します。

2. (プロセス モニタリングのみ) エントリ内で定義された正規表現を使用して、エージェントがモニタ対象のプロセス ID (または Windows サービス インデックス) を判断します。

3. エージェントは指定されたポーリング間隔に従って、指定された MIB 属性 (OID) またはプロセス属性をポーリングします。
4. 属性がしきい値を超えると、エージェントはモニタ エントリの現在のステータスを設定します。現在のステータスが、エントリの重大度列に定義されている値に設定されます。

注: エージェントは、同じオブジェクト情報、`monObjClass`、`monObjInstance`、`monObjAttribute` について同じ値を持つモニタの状態を集計します。モニタ集計の詳細については、「状態管理と集計の仕組み」を参照してください。

5. エージェントは、トラップ コミュニティ内のすべてのシステムに状態変更トラップを送信します。トラップには、オブジェクト情報、超過したしきい値、および現在のオブジェクト状態（集計された全モニタの最悪の状態）が示されます。
6. エージェントは、テーブル エントリ（最悪の状態であるモニタ対象オブジェクトの）で定義されている任意のアクションを実行します。たとえば、停止プロセスに対してプロセス再起動アクションが定義されている場合、エージェントはプロセスを再起動します。
7. エージェントは属性のポーリングを続け、しきい値条件が成立しなくなるとモニタの現在のステータス値を更新し、状態変更トラップを送信します（最悪のステータスが変化する場合）。

レガシー モニタを使ったエージェント操作の詳細については、「[レガシー モニタ \(P. 80\)](#)」を参照してください。

ステートレス モニタリングの仕組み

ステートレス モニタは、オブジェクト ステータス情報を導出せず、オブジェクト モデルを使って全体的なオブジェクト状態を維持することはありません。このようなモニタは重大度の値を維持しますが、この重大度は個別のモニタの重要度を追跡するためのもので、オブジェクト状態の計算には使用されません。以下のテーブルがステートレス モニタリングをサポートしています。

- [プロセス グループ モニタ](#) (P. 65)
- [ログ ファイル モニタ](#) (P. 66)
- [NT イベント モニタ](#) (P. 67)

プロセス グループ モニタ テーブル内にエントリを作成するときは、変更についてモニタし、グループ統計を計算するための一連のプロセスを指定します。ログ ファイル モニタ テーブル内にエントリを作成するときは、正規表現についてモニタするログ ファイルを指定します。ログ ファイル モニタ テーブルでは、エントリの数とサイズを計算するためのディレクトリを追加で指定できます。NT イベント モニタ テーブル内にエントリを作成するときは、特定タイプのイベントについて特定の Windows イベント ログをモニタするための正規表現を指定します。

注: プロセス グループ、ログ ファイル、および Windows イベント モニタリングに関する具体的な情報については、各タイプのモニタリングに関するセクションを参照してください。

SystemEDGE エージェントがステートレス モニタリングの指示を遂行する仕組みを以下に示します。

1. プロセス グループ、ログ ファイル、または Windows イベント ログ内で、指定されたグループ変更、正規表現、またはイベントをモニタするため、ユーザが適切な Systems Management Empire MIB テーブル内にエントリを作成します。

注: ステートレス モニタのエントリ作成の詳細については、「プロセス グループ モニタリング」、「ログ ファイル モニタリング」、および「Windows イベント モニタリング」の各章を参照してください。

CA Virtual Assurance を使用していない限り、sysedge.cf ファイルへの変更を認識させるには、エージェントを再起動する必要があります。エントリを作成し、CA Virtual Assurance を通じて変更を展開すると、エージェントはウォームスタートを実行し、起動中にエントリの読み取りを開始します。

2. 指定されたポーリング間隔に従って、エージェントがエントリを読み取り、システム、指定されたログ ファイル、または指定された Windows イベント ログ内で、指定された正規表現に一致するプロセス、エントリ、またはイベントを検索します。
3. (プロセス グループ モニタリングのみ) エージェントは、正規表現およびモニタ対象プロセスの PID に一致するプロセスの数をプロセス グループ モニタ テーブル エントリに入力します。また、プロセスをポーリングし、集計するグループ統計が含まれるプロセス グループ モニタ エントリの読み取り専用属性の値を更新して、グループ メンバシップの変更をチェックします。
4. エージェントは、プロセス グループの変更、一致するログ ファイル メッセージ、一致するイベントを検出すると、トラップ コミュニティ内のすべてのシステムにトラップを送信します。
5. エージェントは、テーブル エントリ内に定義されたすべてのアクションを実行します。
6. エージェントは、プロセス グループ、ログ ファイル、ディレクトリ、または Windows イベント ログでトラップ条件のポーリングを続けます。

セルフ モニタリング

SystemEDGE エージェントは、サポート対象の MIB または AIM 実装に登録されているタイプ (INTEGER、GAUGE、COUNTER および TIMETICKS) の属性をモニタできます。また、ユーザ定義のしきい値についてモニタし、しきい値違反があればトラップを使って管理システムに報告できます。セルフ モニタ テーブルによるモニタリングでは、モニタの対象、報告する例外条件、および各条件の重大度を設定できます。

セルフ モニタ テーブル内にエントリを作成するときは、ポーリング間隔、比較演算子、しきい値、および重大度を設定します。エージェントは、指定された MIB オブジェクトを自動的にモニタします。MIB オブジェクトおよび MIB 属性という言葉は同じ意味で使用されます。エージェントは、指定された重大度値を使ってオブジェクトのステータスを判断します。また、定義された複数のモニタを使用して、オブジェクトのステータスを集計することができます。

たとえば、特定のファイルシステム上の使用済み領域をモニタするために SystemEDGE を設定できます。複数の重大度を使ってしきい値をモニタするには、以下のように `sysedge.cf` 内に複数のエントリを作成します。

```
monitor devTableEntry 'C:' devCapacity 30 0x0 30 absolute >= 80 'C:' '' 'SysHealth'
'sysdrive' 'FS usage' 'ok'
```

```
monitor devTableEntry 'C:' devCapacity 30 0x0 120 absolute >= 80 'Used space on C:
drive' '' 'SysHealth' 'sysdrive' 'FS usage' 'warning'
```

```
monitor devTableEntry 'C:' devCapacity 31 0x0 120 absolute >= 85 'Used space on C:
drive' '' 'SysHealth' 'sysdrive' 'FS usage' 'minor'
```

```
monitor devTableEntry 'C:' devCapacity 32 0x0 120 absolute >= 90 'Used space on C:
drive' '' 'SysHealth' 'sysdrive' 'FS usage' 'critical'
```

この例では、しきい値ごとにさまざまな重大度を使って複数のエンティティを定義しています。モニタ対象のオブジェクトは、パラメータ クラス (SysHealth)、インスタンス (sysdrive)、および属性 (FS usage) によって定義されます。オブジェクトのステータスは、OK (どのしきい値も超えていない)、マイナー、警告、または重大 (対応するしきい値で違反が発生している) のいずれかです。オブジェクト ステータスが変わるたびに、エージェントは状態変更トラップを送信します。

注: 詳細および具体的なモニタリング例については、「[セルフ モニタリング](#) (P. 231)」の章を参照してください。

関連項目:

[ステートフル モニタリングの仕組み](#) (P. 53)

サポートされている MIBs

セルフ モニタ テーブル内にエントリを作成するときは、サポートされている任意の MIB 内で指定のしきい値について属性をモニタするようエージェントに指示できます。

- CA Systems Management Empire MIB
- MIB-II (RFC 1213)
- Host Resources MIB (RFC 1514)
- IP-MIB (部分的) (RFC 4293)
- IF-MIB (部分的) (RFC 2233)
- TCP-MIB (部分的) (RFC 4022)
- UDP-MIB (部分的) (RFC 4113)
- SystemEDGE バージョン 5.0 AIM の任意の MIB

Host Resources MIB 内のオブジェクトの詳細については、付録「Host Resources MIB」を参照してください。MIB-II 内のオブジェクトの詳細については、RFC 1213 (英語) を参照してください。

Systems Management Empire MIB グループ

Systems Management Empire MIB は、セルフ モニタリング用のオブジェクトのテーブルとグループが含まれる、企業に固有の属性コレクションです。

以下のグループとテーブルには、Systems Management Empire MIB でモニタできるオブジェクトが含まれます。

システム グループ (sysedgeSystem)

基本的なシステム情報が含まれます (ホスト名、CPU タイプ、メモリの量、タイムゾーン、システムの完全修飾ドメイン名、ロードされたプラグインのリスト、SystemEDGE の稼働時間など)。

マウントされたデバイス テーブル (devTable)

ホスト上にマウントされたデバイスとファイル システムに関する情報が含まれます (マウント ポイント、ブロック サイズ、空きブロック、使用中の iNode、ファイル名の最大長など)。また、ファイル システムの使用状況などの値についてモニタを作成したり、テーブル内に列値を設定することでマウント済みデバイスをマウント解除したりできます。

カーネル設定グループ (kernelConfig)

カーネル情報が含まれます (CPU の数、仮想メモリの量、プロセスあたりのメモリ、プロセスあたりのオープン ファイル、スワップ領域、クロック速度など)。カーネルのプロパティと設定をモニタするには、このグループを使用します。

ブート設定グループ (bootconf)

ルート ファイル システム、ダンプ ファイル システム、およびスワップ領域に関する情報が含まれます。ルート ファイル システム名、ファイル システム ブロック、ファイル システム タイプなどの値を追跡するには、このテーブルをモニタします。

ストリーム グループ (streams)

ストリーム I/O サブシステムに関する情報が含まれます。使用中のストリーム数、ストリーム割り当て失敗の数、キュー内のストリーム数などのオブジェクトをモニタしてサブシステムのヘルスを追跡するには、このグループを使用します。

ユーザ テーブル (userTable)

システム上のユーザ アカウントに関する情報が含まれます。

グループ テーブル (groupTable)

システム上のユーザ グループに関する情報が含まれます。

プロセス テーブル (processTable)

実行中のプロセスに関する情報が含まれます。現在実行中のプロセスを追跡し、プロセス制御のための特定の属性を設定するには、このテーブルをモニタします。たとえば、プロセスを停止するには、**processkill** 列の値を **9** に設定します。

Who テーブル (whoTable)

システムに現在ログオンしているユーザに関する情報が含まれます。ある特定の時間にシステムを使用しているユーザを追跡するには、このテーブル内の属性をモニタします。

リモートシェル グループ (remoteshell)

リモート システム上でシェル スクリプトとプログラムを実行するための属性が含まれます。コマンド、その引数、および出力ファイルの名前を指定するには、このテーブル内の属性を設定します。

カーネル パフォーマンス グループ (kernelperf)

ホスト オペレーティング システムのヘルスおよびパフォーマンスに関する情報が含まれます。現在のプロセスとオープン ファイルの数、アクティブなジョブの数、スケジューラの実行キュー内にあるジョブの数などの属性をモニタできます。

プロセス間通信テーブル (msgqueueTable、shmemTable、semTable)

メッセージ キュー、共有メモリ、およびセマフォに関する情報が個別のテーブルに含まれます。プロセス間の通信を調整するには、これらのテーブルをモニタします。

メッセージ バッファ割り当てテーブル (mbufAllocTable)

システムでのメッセージ バッファの使用状況に関する情報が含まれます。バッファ要求が拒否または遅延された回数などの情報を追跡するには、このテーブル内の属性をモニタします。

ストリーム バッファ割り当てテーブル (strbufAllocTable)

ストリーム サブシステムによって使用されるバッファのバッファ割り当ておよび使用状況の統計に関する情報が含まれます。

I/O バッファ キャッシュ グループ (ioBufferCache)

ベーシック ディスク I/O の I/O バッファ割り当ておよび使用状況に関する情報が含まれます。I/O バッファ アクティビティのピーク時などの情報を追跡するには、このテーブルをモニタします。

ディレクトリ名のルックアップ キャッシュ グループ (dnlc)

ディレクトリとファイルの名前キャッシュのパフォーマンスに関する情報が含まれます。

AIX 論理パーティション グループ(logicalPartition)

IBM AIX 論理パーティション (LPAR) に関する情報が含まれます。各パーティションの物理または論理 **CPU**、各パーティションの **CPU** の数などの属性をモニタできます。

トラップ コミュニティ テーブル(trapCommunityTable)

設定されたコミュニティ、ユーザ、トラップ先などの **SNMP** 情報が含まれます。

NT システム グループ(ntSystem)

Windows システムに固有の情報が含まれます。このグループには、**Windows** システム上の各種領域の属性をモニタするための、システム、スレッド、レジストリ、サービス、システム パフォーマンス、キャッシュ パフォーマンス、メモリ パフォーマンス、ページファイル パフォーマンス、およびイベント モニタの各グループが含まれます。

RPC 統計グループ(rpc)

カーネルのリモート プロシージャ コールに関する情報が含まれます。**RPC** アクティビティのピーク時を検出するためのカウンタや統計などの属性を追跡するには、このテーブルをモニタします。

NFS 統計グループ(nfs)

カーネルの **NFS** 機能に関する情報が含まれます。**NFS** アクティビティのピーク時を検出するための統計やカウンタなどの属性を追跡するには、このテーブルをモニタします。

ディスク統計テーブル(diskStatsTable)

ディスク **I/O** に関する情報が含まれます。

CPU 統計テーブル(cpuStatsTable)

各 **CPU** のパフォーマンス統計が含まれます。アイドルモードの時間や待機モードの時間などの属性を追跡するには、このテーブルをモニタします。

Systems Management Empire MIB には、オブジェクト集計をサポートするモニタリング テーブルやテーブルも含まれます。オブジェクト集計の詳細については、「[状態管理モデル](#) (P. 70)」を参照してください。

セルフ モニタのトラップ

SystemEDGE エージェントは、セルフ モニタ テーブル内の重大度が **none(1)** を超えるエントリをモニタするとき、以下のトラップを送信します。

aggregateStateTrap

モニタの状態が変わったことを示すため、エージェントはこのトラップを送信します。同じオブジェクト情報を含むモニタが集計されます。対応する集計状態は、このような集計モニタの最悪の状態と等しくなります。

たとえば、重大度が「重大」であるモニタでしきい値違反が発生しても、集計モニタの現在の状態は「致命的」であるとします（重大度が「致命的」である、同じ属性のための別のモニタでしきい値違反が発生したことを意味します）。この場合、集計状態は影響を受けないので、エージェントはトラップを送信しません。

注: モニタ集計トラップおよび他の集計トラップの詳細については、「[状態管理モデル](#) (P. 70)」を参照してください。

SystemEDGE エージェントは、セルフ モニタ テーブル内の重大度が「なし」（レガシー モニタ）のエントリをモニタするとき、以下のトラップを送信します。

monitorTrap

モニタ対象属性のしきい値条件で違反が発生すると、エージェントはこのトラップを送信します。

monitorEntryNotReadyTrap

セルフ モニタ テーブルエントリの行ステータスが **notReady(3)** に設定されているとき、エージェントはこのトラップを送信します。**notReady** 行ステータスは、オブジェクトがアクセス不可能であることを示す場合があります。

monitorClearTrap

以前存在していたしきい値違反がなくなったことを示すため、エージェントはこのトラップを送信します。

エントリ内で重大度を定義している場合、レガシー モニタのトラップは送信されません。レガシー モニタの詳細について、およびレガシー モードでのエージェント使用の詳細については、「[レガシー モニタ](#) (P. 80)」を参照してください。

これらのトラップおよび他のトラップの詳細については、付録「プライベート企業トラップ」を参照してください。

プロセスとサービスのモニタリング

SystemEDGE エージェントは、任意のプロセスまたは Windows サービスをモニタして、その実行ステータスやプロセス属性の値をチェックできます。モニタするプロセスおよび追跡する属性を設定するには、Systems Management Empire MIB のプロセス モニタ テーブル内にエントリを作成します。

プロセス モニタ テーブル内にエントリを作成するときは、ポーリング間隔、モニタするプロセス（またはサービス）を表す正規表現、比較演算子、しきい値、および重大度を設定します。エージェントは、プロセス ID を自動的に判断し、指定されたプロセス属性をモニタします。エージェントは、指定されたしきい値と重大度値を使ってプロセス属性のステータスを判断します。また、定義された複数のモニタの状態を集計して、全体的なオブジェクト状態を導出できます。

たとえば、ミッションクリティカルなプロセスにおいてシステム コールをモニタすることができます。指定されたしきい値を超えると、エージェントはオブジェクトの重大度を調整し、状態変更トラップを送信します。プロセスが実行されているかどうかをモニタすることもできます。エージェントは停止されたプロセスを自動的に再起動できます。

注: 詳細については、「プロセスとサービスのモニタリング」の章を参照してください。

関連項目:

[ステートフル モニタリングの仕組み](#) (P. 53)

プロセス モニタのトラップ

SystemEDGE エージェントは、プロセス モニタ テーブル内の重大度が none(1) を超えるエントリをモニタするとき、以下のトラップを送信します。

aggregateStateTrap

プロセス モニタの状態が変わったことを示すため、エージェントはこのトラップを送信します。このトラップを送信するのは、しきい値条件違反によってプロセス モニタの状態が変わる場合のみです。たとえば、重大度が「重大」であるモニタでしきい値違反が発生しても、集計モニタの現在の状態は「致命的」であるとします（重大度が「致命的」である、同じ属性のための別のモニタでしきい値違反が発生したことを意味します）。この場合、集計状態は影響を受けないので、エージェントはトラップを送信しません。

注: モニタ集計トラップおよび他の集計トラップの詳細については、「[状態管理モデル](#) (P. 70)」を参照してください。

SystemEDGE エージェントは、プロセス モニタ テーブル内の重大度が「なし」（レガシー モニタ）のエントリをモニタするとき、以下のトラップを送信します。

processThresholdTrap

モニタ対象プロセス属性のしきい値条件で違反が発生すると、エージェントはこのトラップを送信します。

processStopTrap

モニタ対象プロセスが停止したか、実行できない状態にあるとき、エージェントはこのトラップを送信します。

processStartTrap

停止していたモニタ対象プロセスが再開したことを示すため、エージェントはこのトラップを送信します。

processClearTrap

以前存在し、トラップ生成を発生させていたしきい値違反がなくなったことを示すため、エージェントはこのトラップを送信します。

processNotReadyTrap

プロセス モニタ テーブル エントリの行ステータスが **notReady(3)** に設定されているとき、エージェントはこのトラップを送信します。
notReady 行ステータスは、プロセスまたは属性がアクセス不可能であることを示す場合があります。

エントリ内で重大度を定義している場合、レガシー モニタのトラップは送信されません。レガシー モニタの詳細について、およびレガシー モードでのエージェント使用の詳細については、「[レガシー モニタ \(P. 80\)](#)」を参照してください。

これらのトラップおよび他のトラップの詳細については、付録「プライベート企業トラップ」を参照してください。

プロセス グループ モニタリング

SystemEDGE エージェントにとって、単一プロセスの属性値を合計し、グループ内のプロセスを追跡するには、一連のプロセスをモニタする追加のセルフ モニタが必要です。プロセスをグループとしてモニタするための設定を行うには、**Systems Management Empire MIB** のプロセス グループ モニタ テーブル内にエントリを作成します（追加のセルフ モニタが必要です）。

プロセス グループ モニタ テーブル内にエントリを作成するときは、ポーリング 間隔、プロセス グループを追加するための正規表現、およびモニタ重大度を設定します。エージェントは、グループとして正規表現に一致するすべてのプロセスをモニタします。エージェントは、プロセス グループ モニタ テーブルの読み取り専用属性に、合計プロセス サイズ、スレッド数、実メモリ使用率などの合計グループ統計値を入力します。プロセス グループのメンバシップに変更があった場合（正規表現に一致する新しいプロセスをエージェントが検出した場合や、グループ内のプロセスがアクセス不可能になった場合など）、エージェントはグループ変更トラップを送信します。**UNIX** では、ユーザ別やグループ名でプロセスを照合するようにエージェントを設定することもできます。

たとえば、重要なプロセスのグループをモニタし、そのうちのいずれかがダウンしたときに通知を受け取るようにすることができます。特定のプロセスがダウンする場合は、プロセス グループ モニタ テーブル内の読み取り専用属性にセルフ モニタを設定できます。たとえば、一連のプロセスによって使用される **CPU** 合計時間をモニタし、対応するアプリケーションに関連付けられているサービス レベル アグリーメントに相当するしきい値を設定できます。

注: 詳細については、「プロセス グループ モニタリング」の章を参照してください。

関連項目:

[ステートレス モニタリングの仕組み](#) (P. 54)

プロセス グループ モニタのトラップ

プロセス グループ モニタ テーブル内のエントリをモニタするとき、SystemEDGE エージェントは以下のトラップを送信します。

procGroupChange

プロセス グループのメンバシップに変更があったことを示すため、エージェントはこのトラップを送信します。変更の理由として、プロセス グループ モニタ エントリの指定に一致する新しいプロセスが検出されたか、以前のモニタ対象エントリがアクセス不可能になったことが挙げられます。モニタの重大度を定義している場合、トラップには重大度の値が含まれます。

このトラップおよび他のトラップの詳細については、付録「プライベート企業トラップ」を参照してください。

ログ ファイル モニタリング

SystemEDGE エージェントは、どのようなテキスト ファイルでも行単位で正規表現をモニタできます。ログ ファイルをモニタするためにエージェントを設定するには、Systems Management Empire MIB のテキスト (ログ) ファイル モニタ テーブル内にエントリを作成します。ログ ファイル モニタリングでは、複数のシステム ログ ファイルまたはアプリケーション ログ ファイル内で、アクションの必要なエントリを自動的にモニタできます。

ログ ファイル モニタ テーブル内にエントリを作成するときは、ポーリング間隔、モニタするログ ファイル (パターン)、モニタ重大度、および検索する正規表現を設定します。エージェントは、ログ ファイル内で指定の正規表現を検索し、表現に一致するメッセージがファイルに含まれる場合にトラップを送信します。

たとえば、アプリケーション ログ ファイル内で、失敗のメッセージまたは未承認の使用/不正な使用を示すメッセージをモニタするために SystemEDGE エージェントを設定できます。

また、ログ ファイル モニタ テーブル内にエントリを作成して、ファイルディレクトリ内でサイズとコンテンツをモニタすることもできます (追加のセルフ モニタが必要です)。

注: 詳細については、「ログ ファイル モニタリング」の章を参照してください。

関連項目:

[ステートレス モニタリングの仕組み](#) (P. 54)

ログ ファイル モニタのトラップ

ログ モニタ テーブル内のエントリをモニタするとき、SystemEDGE エージェントは以下のトラップを送信します。

logMonMatchTrap

指定された正規表現の一致をログ ファイル内で検出したことを示すため、エージェントはこのトラップを送信します。モニタの重大度を定義している場合、トラップには重大度の値が含まれます。

logMonNotReadyTrap

ログ モニタ テーブルエントリの行ステータスが **notReady(3)** に設定されているとき、エージェントはこのトラップを送信します。**notReady** 行ステータスは、ファイルがアクセス不可能であることを示す場合があります。

これらのトラップおよび他のトラップの詳細については、付録「プライベート企業トラップ」を参照してください。

Windows イベント モニタリング

SystemEDGE エージェントは、Windows イベント ログ内で、重要なイベント タイプ、イベント識別子、または特定の正規表現に一致するイベントをモニタできます。Windows イベント ログをモニタするためにエージェントを設定するには、Systems Management Empire MIB の NT イベント モニタ テーブル内にエントリを作成します。Windows イベント モニタリング エントリを作成すると、重大なイベントまたはアクションの必要なイベント用の Windows イベント ログを管理できます。Windows イベントにはテキストのメッセージに加えていくつかの識別できる特徴が含まれるので、このモニタリング機能を使用することで、基本的なログ ファイル モニタリングよりも高度なタイプの一致条件を指定できます。

NT イベント モニタ テーブル内にエントリを作成すると、NT イベント モニタは一定の間隔でポーリングを行います。モニタするイベント タイプとソース、モニタ重大度、イベント メッセージテキストに対して照合する正規表現（イベント ID を含む）がポーリングされます。エージェントは Windows イベント ログをモニタし、表現に一致するイベントが検出されるとトラップを送信します。

たとえば、Windows Security イベント ログ内で認証の失敗をモニタするために SystemEDGE エージェントを設定できます。

注: 詳細については、「Windows イベント モニタリング」の章を参照してください。

関連項目:

[ステートレス モニタリングの仕組み](#) (P. 54)

Windows イベント モニタのトラップ

NT イベント モニタ テーブル内のエントリをモニタするとき、SystemEDGE エージェントは以下のトラップを送信します。

ntEventMonMatchTrap

指定された正規表現の一致を **Windows** イベント ログ内で検出したことを示すため、エージェントはこのトラップを送信します。モニタの重大度を定義している場合、トラップには重大度の値が含まれます。

ntEventMonNotReadyTrap

NT イベント モニタ テーブル エントリの行ステータスが **notReady(3)** に設定されているとき、エージェントはこのトラップを送信します。**notReady** 行ステータスは、**Windows** イベント ログ ストアがアクセス不可能であることを示す場合があります。

これらのトラップおよび他のトラップの詳細については、付録「プライベート企業トラップ」を参照してください。

履歴の収集

SystemEDGE エージェントは、サポートされている任意の MIB の属性値（メトリック）をサンプリングできます。収集されたサンプルを使用すれば、システムのベースライニングおよび傾向分析を行うことができます。サンプルは履歴テーブルに入力されます。オプションで、サンプルをキューブ ファイルに書き込むこともできます。

履歴の収集を設定するには、Systems Management Empire MIB の履歴制御テーブル内にエントリを作成します。履歴の収集を設定すると、マネージャを常時ポーリングしなくても、ある一定の期間にわたってメトリックを分析できます。キューブ ファイルに書き込まれるデータは、CA Virtual Assurance Systems Performance による分析に利用できます。

履歴制御テーブル内にエントリを作成するときは、ポーリング間隔、収集する MIB 属性、および保存するサンプルの数を設定します。SystemEDGE は設定のとおりデータを収集します。マネージャは定期的に履歴を取得することで、一定の期間にわたって属性を表示できます。

たとえば、Systems Management Empire MIB の CPU 統計テーブルから CPU MIB 属性を 10 分おきに収集し、最後の 288 個のサンプルを保存するようにエージェントを設定できます。次に、収集済みのデータを分析用に取得するため、48 時間おきにデータをアップロードするように管理システムを設定できます。

履歴の収集の設定の詳細については、「履歴の収集」の章を参照してください。

履歴の収集の仕組み

SystemEDGE エージェントが履歴の収集の指示を遂行する仕組みを以下に示します。

1. 指定された期間にわたって MIB 属性をモニタするため、ユーザが Systems Management Empire MIB の履歴制御テーブル内にエントリを作成します。

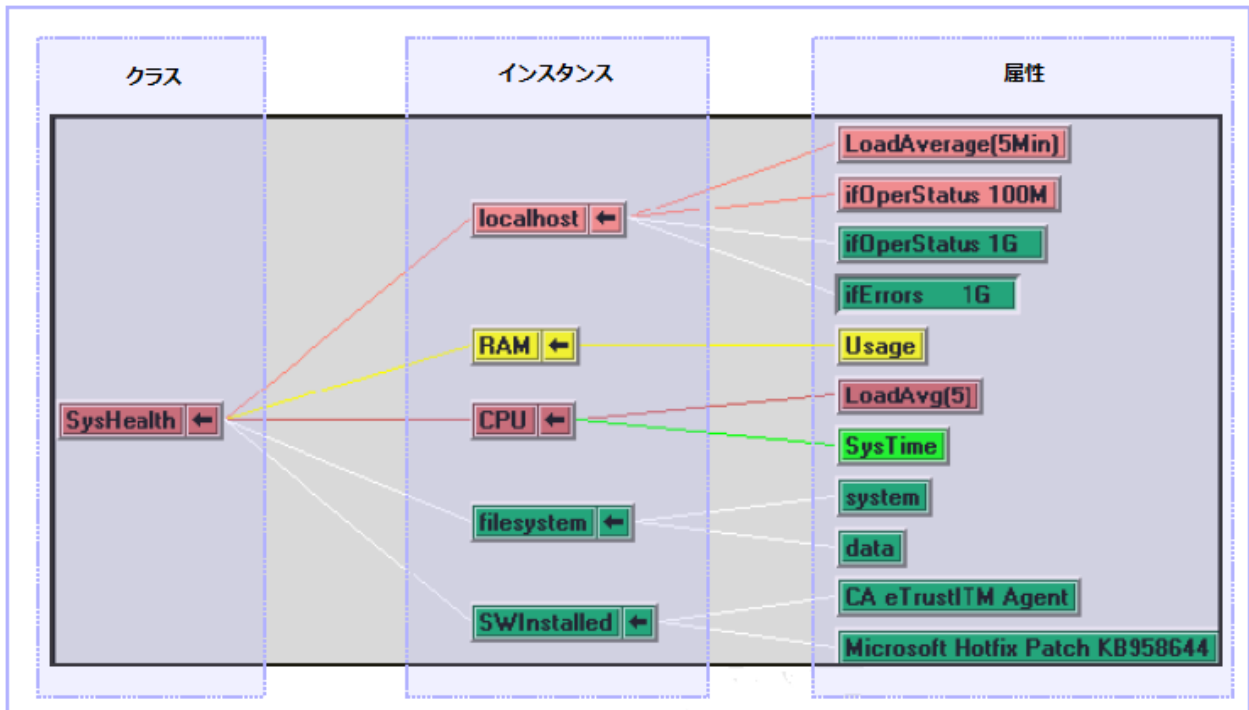
注: 履歴の収集エントリ作成の詳細については、「履歴の収集」の章を参照してください。

CA Virtual Assurance を使用していない限り、sysedge.cf ファイルへの変更を認識させるには、エージェントを再起動する必要があります。エントリを作成し、CA Virtual Assurance を通じて変更を展開すると、エージェントはウォームスタートを実行し、起動中にエントリの読み取りを開始します。

2. エージェントが、指定されたポーリング間隔で属性をポーリングし、履歴テーブル（スライディング ウィンドウ）の中に属性情報を収集します。指定のサンプル数に到達すると、最も古いサンプルが削除されます。

状態管理モデル

SystemEDGE エージェントは、CA Virtual Assurance 管理モデル全体と完全に統合された、セルフ モニタおよびプロセス モニタのための状態管理モデルをサポートしています。エージェントは重要度の異なる複数のモニタを単一の管理対象オブジェクトに集計します。このオブジェクトは、最悪の重大度を持つモニタに相当する状態を持ちます。以下の図に、この新しいメカニズムによって提供される柔軟性と機能を示します。



図中の「属性」ペインには複数のモニタ対象オブジェクト

(SysHealth.localhost.LoadAverage など) が含まれます。SystemEDGE エージェントがオブジェクトに割り当てることができる状態は、CIM ManagedSystemElement Management の状態に準拠しています。各属性オブジェクトの状態は、対応する SystemEDGE モニタによって定義されます。たとえば、重大度が「致命的」である LoadAverage しきい値 250 を割り当て、現在の LoadAverage 値が 250 である場合、モニタ対象オブジェクトの状態は「致命的」です。

SystemEDGE の柔軟なオブジェクトモデルによって、さまざまなソースからの異種オブジェクトをグループ化することが可能になります。オブジェクトの状態は各グループ全体で集計されます。たとえば、エージェントはサーバ CPU のヘルス状態を、Web ページにアクセスするのににかかった応答時間 (SRM AIM によって収集) やディスクのパフォーマンス (リモート モニタリング AIM によって収集) と共に集計することができます。

図中の「クラス」ペインは、状態管理モデルによって管理対象オブジェクトがどのように集計管理対象オブジェクトにグループ化されるかを示しています。図に示されているとおり、SysHealth オブジェクトはそのすべての管理対象オブジェクトの状態を集計します。

状態管理の機能は、Systems Management Empire MIB のセルフ モニタ テーブルとプロセス モニタ テーブルから設定することができます。オブジェクトおよび重大度情報を含む列は、各モニタ エントリのオブジェクト ステータスを計算およびプロパゲートするのに役立ちます。

オブジェクト ステータス

セルフ モニタ テーブルとプロセス モニタ テーブルに入力される重大度値によって、モニタ対象オブジェクトのオブジェクト ステータスが決まります。特定のしきい値条件に違反するモニタ対象オブジェクトには特定のステータスが割り当てられます。他の 3 つのモニタリングテーブル内に入力される重大度値では、モニタ対象オブジェクトのステータスではなく、モニタに関連付けられているイベントの重要度が決まります（ステートレス モニタリング）。

モニタに対して入力できる重大度値を以下に示します。

- none(1)
- ok(2)
- warning(3)
- minor(4)
- major(5)
- critical(6)
- fatal(7)

none(1) の重大度を指定すると、レガシー モニタが作成されます。このモニタは、状態変更トラップを送信せずにオブジェクトをモニタします。「なし」は、objectState 列に値を指定しない場合またはモニタを以前のリリースからアップグレードする場合のデフォルトの重大度です。詳細については、「[レガシー モニタ \(P. 80\)](#)」を参照してください。

ok(2) は一時的な「モニタしない」機能として使用できます。現在の状態が ok(2) のとき、モニタ対象属性で重大度違反が発生しても、状態変更トラップは送信されません。

エージェントによって計算され、モニタ エントリの読み取り専用属性として記録される結果のオブジェクト状態は、以下のいずれかになります。

- unknown(1)
- ok(2)
- warning(3)
- minor(4)
- major(5)
- critical(6)
- fatal(7)
- up(11)
- down(12)

モニタの重大度が「なし」の場合、状態は **up(11)** と **down(12)** の間で切り替えられます。

エージェントがシャット ダウンするときは、全モニタの現在の状態が保存されます。これらの保存された状態は、エージェントの起動中にモニタ状態を初期化するのに使用されます。

オブジェクト集計

セルフ モニタ テーブルとプロセス モニタ テーブル内のオブジェクト クラス、インスタンス、および属性値は、モニタ対象（または管理対象）オブジェクトを記述します。オブジェクト インスタンス、クラス、および属性値が同じであるモニタが複数ある場合、そのようなモニタは、接続済みであると見なされ、同じ管理対象オブジェクトを記述します。エージェントは全接続済みモニタの最悪の状態を、1つの管理対象オブジェクト状態に集計します。また、集計テーブル内の集計ごとにエントリ（集計）を作成します。

ただし、2つの接続済みモニタ **A** と **B** が同じ重大度を持ち、「AND」フラグを使用している場合（**monFlags** の 00800 または **pmonFlags** の 20000）、**A** と **B** は同じ状態を持たなければ最悪状態の計算から除外されます。対応する列に値を入力しない場合、意味のあるデフォルト情報がエージェントによって入力されます。エージェントはモニタ対象 **OID** に基づいてデフォルトのセルフ モニタ値を導出し、それらの値をクラス、インスタンス、および属性値にマップします。また、プロセス正規表現およびモニタ対象属性に基づいてデフォルトのプロセス モニタ値を導出します。デフォルトのオブジェクト情報割当ての詳細については、「セルフ モニタリング」および「プロセスとサービスのモニタリング」の章を参照してください。

接続済みモニタはすべて同じ間隔を持つ必要があります。そうすることで、すべての接続済みモニタについて集計状態の評価が1回のみ行われ、不要な状態変更が発生するのを防ぐのに役立ちます。

集計テーブルには以下の情報が含まれます。

- オブジェクトクラス、インスタンス、および属性値
- 行ステータス
- 最悪の集計状態を持つモニタ エントリの値（しきい値、演算子、現在値など）
- 現在の集計状態
- モニタ統計（送信された状態変更トラップの数、任意の状態であった時間、以前のステータスなど）

注: 集計テーブルの列の詳細については、「セルフ モニタリング」の章を参照してください。

エージェントは、集計テーブル内の現在の状態を更新します。エージェントが状態変更トラップを送信するのは、すべての接続済みモニタのしきい値評価およびその後の状態集計の結果、最悪状態がその前の状態から変わったときに限られます。たとえば、3つのモニタを使ってCPU使用率をモニタしているとき、エージェントからCPU使用率82%が返されたと仮定します。使用しているモニタのしきい値は、1つは60%（重大度は「警告」）、1つは80%（重大度は「重大」）、1つは100%（重大度は「致命的」）です。返された値の場合、60%および80%のモニタでしきい値違反が生じますが、エージェントは80%のモニタについてのみ1つの状態変更トラップを送信し、集計状態を「重大」に変更します。

スーパー集計

集計テーブルには、オブジェクトクラス、インスタンス、および属性が同じで、重大度（しきい値）が異なる、すべてのセルフ モニタおよびプロセス モニタのエントリがデフォルトで含まれます。このテーブルは重大度レベル（標準集計）で集計を行います。

また、このテーブルでは、クラス レベル、属性レベル、またはインスタンス レベルでのスーパー集計も可能です。対応するテーブルエントリは、プレースホルダとしてアスタリスク（*）を使用します。

属性レベル

オブジェクトクラスとインスタンスが同じであるすべてのモニタについて 1 つのエントリ（`aggObjAttribute` を * に設定）

インスタンスレベル

オブジェクトクラスと属性が同じであるすべてのモニタについて 1 つのエントリ（`aggObjInstance` を * に設定）

クラスレベル

オブジェクトインスタンスと属性が同じであるすべてのモニタについて 1 つのエントリ（`aggObjClass` を * に設定）。

上記の 3 レベルの組み合わせは `sysedge.cf` ファイルで設定でき、全体的なエージェント状態までの集計が可能になります。

スーパー集計のインデックスは 100001 から始まります。

重要: *, *, * エントリの行ステータス属性を `destroy(6)` に設定すると、`none(1)` より大きい重大度を持つすべてのモニタは事実上削除されます。

集計レベルは、以下の形式に従って `sysedge.cf` ファイル内で設定できます。

`aggregate_level` レベル

level

以下の集計レベルの指定を可能にするビット フィールドを指定します。

001 : クラスとインスタンスが同じすべてのモニタを、属性が * に設定された 1 つのエントリに集計

002 : クラスと属性が同じすべてのモニタを、インスタンスが * に設定された 1 つのエントリに集計

004 : インスタンスと属性が同じすべてのモニタを、クラスが * に設定された 1 つのエントリに集計

010 : クラスが同じすべてのモニタを、インスタンスと属性が * に設定された 1 つのエントリに集計

020 : インスタンスが同じすべてのモニタを、クラスと属性が * に設定された 1 つのエントリに集計

040 : 属性が同じすべてのモニタを、クラスとインスタンスが * に設定された 1 つのエントリに集計

100 : すべてのモニタを、クラス、インスタンス、属性が * に設定された 1 つのエントリに集計

集計テーブルのエントリ（および一部の選択された属性）は、`sysedge.mon` ファイルに保存されます。その主な目的は、再起動間でエントリのインデックスを保存することです。マネージャは通常、インデックスを使ってオブジェクトのポーリングを行い、インデックスが頻繁に変更されないことを前提としているためです。

状態管理と集計の仕組み

このセクションで説明するとおり、SystemEDGE エージェントは、セルフ モニタリングとプロセス モニタリングの指示を実行しながら、オブジェクト ステータスを計算および集計します。

注: このプロセスは、状態管理と集計のデフォルトの動作を表しています。ステータスの管理とオブジェクト ステータスの集計を異なる方法で行うためのエージェント設定の詳細については、「[状態管理の設定オプション \(P. 79\)](#)」を参照してください。

1. ユーザが、重大度が「なし」を超えるセルフ モニタ エントリまたはプロセス モニタ エントリを、セルフ モニタ テーブルまたはプロセス モニタ テーブル内に作成します。オブジェクト インスタンス、クラス、および属性情報は定義すべきです。定義しない場合、エージェントによって、意味のあるデフォルト情報がこれらの列に入力されます。

CA Virtual Assurance を使用していない限り、sysedge.cf ファイルへの変更を認識させるには、エージェントを再起動する必要があります。エントリを作成し、CA Virtual Assurance を通じて変更を展開すると、エージェントはウォームスタートを実行し、起動中にエントリの読み取りを開始します。

2. エージェントが、指定されたポーリング間隔に従って、エントリ内に定義された属性をモニタし、指定された行ステータス、しきい値違反条件、および重大度に従って、モニタの現在の状態を導出します。また、モニタ エントリの読み取り専用属性に現在の状態を入力します。
3. エージェントが集計テーブルに状態情報を追加する方法は以下のとおりです。
 - モニタのオブジェクト クラス、インスタンス、および属性値を持つエントリが集計テーブル内にすでに存在する場合。エージェントは（内部的に）モニタをテーブル内のエントリに接続します。
 - モニタのオブジェクト クラス、インスタンス、および属性値を持つエントリが集計テーブル内に存在せず、エージェントが集計テーブルに新しいエントリを追加する場合。
4. エージェントは、集計テーブルに新しいエントリを追加するとき、トラップ コミュニティ内のすべてのシステムに集計追加トラップを送信します。
5. モニタがしきい値を超えると、エージェントはセルフ モニタ テーブルまたはプロセス モニタ テーブル内でモニタの現在の状態を変更します。次に、全接続済みモニタの最悪の状態を集計し、集計テーブル内の対応するエントリを更新します。

6. エージェントは集計の状態が変わった場合のみ状態変更トラップを送信します。

注: 集計は指定された遅延設定に基づいて行われます。モニタ エントリに対して遅延値を設定した場合、遅延時間の条件が満たされるまで、個々のモニタ状態変更または集計状態変更は発生しません。テーブルフラグを使った遅延値設定の詳細については、「セルフ モニタリング」および「プロセスとサービスのモニタリング」の章を参照してください。

7. エージェントは、最悪状態のモニタに関連付けられたアクションを実行します。
8. エージェントは、属性のポーリングを続行し、以下の状況では説明のとおりに動作します。

- しきい値違反条件が成立する場合、エージェントはモニタに関連付けられたアクションの実行を続けますが（最悪状態のモニタを表している限り）、追加の状態変更トラップを送信することはありません。
- しきい値違反条件が成立しなくなると、エージェントはモニタ内の現在の状態を更新し、集計テーブル内の最悪の状態を現在の最悪の状態を更新します。次に、適切な状態変更トラップを送信します。
- より大きな重大度を持つ接続済みモニタがしきい値を超えた場合、エージェントは集計状態を変更し、状態変更トラップを送信します。

注: エージェントは、セルフ モニタ テーブルおよびプロセス モニタ テーブル内の個々のセルフ モニタ エントリおよびプロセス モニタ エントリの管理を継続します。エージェントは個々のモニタで属性のポーリングを継続し、各モニタの状態をテーブル内に維持します。このような理由により、すべての接続済みモニタには同じポーリング間隔を設定すべきです。そうすれば、すべてのオブジェクトモニタが同時に評価され、不必要な状態変更を防ぐことができます（たとえば、ポーリング間隔が不統一である場合、「OK」から「重大」に変わり、その後に「重大」から「致命的」に変わる可能性があります）。

状態管理トラップ

SystemEDGE エージェントは、重大度が「なし」を超えるセルフ モニタ エントリおよびプロセス モニタ エントリをモニタするとき、以下のトラップを送信します。

注: 重大度が「なし」（レガシー モニタ）のセルフ モニタ エントリおよびプロセス モニタ エントリの場合、エージェントは前のエージェント リリースのしきい値違反トラップを代わりに送信します。このようなトラップの詳細については、「セルフ モニタのトラップ」および「プロセス モニタのトラップ」を参照してください。

aggregateStateTrap

集計の状態が変わるとき、エージェントはこのトラップを送信します（接続済みモニタのしきい値違反によって集計状態が変わる場合）。たとえば、重大度が「重大」であるモニタがしきい値を超えても、現在の集計状態が「致命的」である場合、エージェントはトラップを送信しません。集計状態は影響を受けず、「致命的」のままです。

注: 集計状態が影響を受けなくても、状態変更トラップが送信されるようにエージェントを設定することはできます。詳細については、「エージェントの設定」の章を参照してください。

aggregateAddTrap

集計テーブルに行が追加されると（重大度が「なし」を超えるモニタが作成されたとき、モニタのオブジェクト情報を含む、対応する集計がまだ存在しない場合）、エージェントはこのトラップを送信します。対応する集計が存在する場合、新しいモニタは接続済みモニタとして集計テーブル内の既存の行に（内部的に）追加され、**aggregateAdd** トラップはトリガされません。

aggregateDeleteTrap

集計テーブルから行が削除されるとき、エージェントはこのトラップを送信します。

状態管理の設定オプション

モニタ集計については以下のような設定が可能です。

- 集計テーブル内の同一のエントリをスーパー集計オブジェクトに集計するようにエージェントを設定できます。これは、クラス レベル、インスタンス レベル、および属性レベルで別々に実行できます。

たとえば、属性レベルで集計（スーパー）するように設定されたエージェントは、オブジェクト クラスとインスタンスが同じで属性が異なるすべての集計の集計テーブル内に追加のエントリを作成します。スーパー集計の属性値はアスタリスク（*）です。この例を使用すると、1つのプロセスのすべてのモニタ対象属性を集計して、そのプロセスで1つの集計状態を計算できます。`sysedge.cf` ファイル内のパラメータを使用すると、このような集計の任意の組み合わせを設定できます。

注：集計レベル設定の詳細および例については、「エージェントの設定」の章を参照してください。

- デフォルトでは、エージェントは集計の状態変更トラップのみを送信します。`sysedge.cf` ファイル内のパラメータを使用すると、すべての接続済みモニタのしきい値違反についてレガシー トラップを送信するようにエージェントを設定できます。
- エージェントはデフォルトで、最悪の状態を持つ接続済みモニタに対してのみアクションを実行します。`sysedge.cf` ファイル内のパラメータを使用すると、すべての接続済みモニタに対してアクションを実行できます。

注：集計トラップおよびアクションの設定の詳細については、「エージェントの設定」の章を参照してください。

- `aggRowStatus` 属性は、集計テーブル内で唯一の読み取り/書き込み属性です。テーブルでは、オブジェクト ステータスの集計と同様に、接続済みモニタの行ステータスが集計されます。行ステータス属性の操作では、すべての接続済みモニタの集計行ステータスを変更できます。行ステータスを `active(1)`、`notInService(2)`、または `destroy(6)` に設定すると、すべての接続済みモニタが同じステータスを継承します。行ステータスを `destroy` に設定すると、集計テーブルからエントリが削除されます。

注：行ステータス集計の詳細および集計テーブルからのエントリ削除の詳細については、「セルフ モニタリング」および「プロセスとサービスのモニタリング」の章を参照してください。

レガシー モードの操作

SystemEDGE はレガシー モードで操作することができます。つまり、その設定を制御する CA Virtual Assurance マネージャがなくても操作できます。レガシー モードで実行されるエージェントは、レガシー モニタに限定されません。

レガシー モニタ

既存のマネージャは、オブジェクト状態モデルを組み込まなくても **SystemEDGE** の属性とプロセスをモニタできます。セルフ モニタ テーブルとプロセス モニタ テーブルの状態管理列はすべてオプションです。エージェントがステータス情報を計算せずに属性とプロセスをモニタするようにしたい場合は、該当のフィールドを空のままにしておきます。

状態を維持および計算しないで動作するセルフ モニタとプロセス モニタは、レガシー モニタです。このようなモニタは、**SystemEDGE** バージョン 5.0 より前のすべてのモニタと同じように機能します。ユーザ モニタまたはプロセス モニタの重大度を入力しない場合、モニタはデフォルトで **none(1)** の重大度を継承します。手動による定義またはデフォルト設定のいずれにかかわらず、重大度が **none(1)** のモニタはすべてレガシー モニタです。レガシー モニタは、現在の状態として「稼働中」または「ダウン」しか持たず、状態変更トラップの代わりにレガシーしきい値トラップを送信します。

注: レガシーしきい値トラップの詳細については、「セルフ モニタのトラップ」および「プロセス モニタのトラップ」を参照してください。

モニタの例

以下の例は、モニタおよび結果の集計状態を示しています。

集計状態 = 重大

モニタ	重大度	AND フラグ	違反済み	状態
M1	警告	×	○	警告
M2	警告	×	×	OK
M3	重大	×	○	重大
M4	重大	×	×	OK

集計状態 = OK

モニタ	重大度	AND フラグ	違反済み	状態
M1	警告	○	○	警告
M2	警告	○	×	OK
M3	重大	○	○	重大
M4	重大	○	×	OK

集計状態 = 警告

モニタ	重大度	AND フラグ	違反済み	状態
M1	警告	○	○	警告
M2	警告	○	○	警告
M3	重大	○	○	重大
M4	重大	○	×	OK

CA Virtual Assurance との統合

CA Virtual Assurance は、SystemEDGE エージェントを使って物理システムと仮想システムを管理するために設計された管理ソリューションです。エージェントは CA Virtual Assurance と緊密に連携して、他のマネージャまたはスタンドアロンエージェントでは提供されない機能を提供します。CA Virtual Assurance との統合により、CA Virtual Assurance ユーザインターフェースから SystemEDGE のデータを表示し、すべての管理対象システムに SystemEDGE をリモートで展開および設定することができます。

エージェントをインストールするときは（CA Virtual Assurance 経由またはスタンドアロンで）、CA Virtual Assurance マネージャからエージェントを表示および管理できるようにするために必要なポートとトラップを指定します。また、インストール中はデータディレクトリとデフォルトポートを指定します。CA Virtual Assurance はデータディレクトリを使用して設定変更を配信します。データディレクトリにはデフォルトポート番号が追加されます。たとえば、デフォルトポート 161 を使用する場合、データディレクトリは C:\Program Files\CA\SystemEDGE\data\port161 となります（Windows の場合）。

プラットフォーム管理モジュール

CA Virtual Assurance マネージャ コンポーネントは、プラットフォーム管理モジュール（PMM）を介した SystemEDGE エージェントとの統合を実現します。SystemEDGE PMM は、すべての管理対象システム上の SystemEDGE エージェントを SNMP でポーリングする Web サービスです。エージェントから受け取ったすべての情報を（MIB オブジェクトの値やトラップなど）CA Virtual Assurance アーキテクチャに適合するプロパティへと変換し、対応する管理対象オブジェクト表現を CA Virtual Assurance データベース内に作成します。SystemEDGE PMM は、CA Virtual Assurance ユーザインターフェースでこのような管理対象オブジェクトを通して SystemEDGE 情報を視覚化します。インストール済みエージェント、システム詳細、既存のモニタ、およびステータス情報をすべて表示でき、ユーザインターフェースから設定の変更を行うことができます。

CA Virtual Assurance 内の視覚化および計測と互換性のある AIM はすべて個別の PMM を持ちます。CA Virtual Assurance の場合、サービス レスポンス モニタ AIM、リモート モニタリング AIM、およびすべての仮想モニタリング AIM の PMM が含まれています。

プラットフォーム管理モジュールの詳細について、およびこのモジュールがエージェント情報を CA Virtual Assurance フレームワーク内で統合する方法の詳細については、CA Virtual Assurance のドキュメントを参照してください。

リモート設定および展開

CA Virtual Assurance は、一元化された場所から、ファイルベースの設定および展開までの SystemEDGE のエンドツーエンドの管理を提供します。ファイルベースの設定によって、以下が実行されます。

- コミュニティ文字列、操作パラメータ、トラップ コミュニティ、AIM ロードなど、Systems Management Empire MIB の読み取り専用領域の設定のような、SNMP マネージャで SNMP SET を使用して実行できない変更を可能にする
- 確認可能な配信を有効にし、生じるセキュリティ脆弱性を SNMP より少なくします。
- 完全なリモート エージェント設定および配信を有効にし、ローカルでの手動ファイル操作の必要性を最小にするか、除去します。

CA Virtual Assurance マネージャは、サポートされている各プラットフォームの SystemEDGE のインストール パッケージを提供します。展開ソリューションを使用すると、認証情報およびインストール パラメータを指定でき、必要な情報をすべて指定すると、展開パッケージを保存して、企業内の管理対象システムに展開できます。

CA Virtual Assurance ユーザ インターフェースから、sysedge.cf ファイルのエージェント設定パラメータをすべて設定し、モニタ テーブルのグラフィカル表示を使用して、自動モニタリング エントリを作成することができます。プロファイルとしてエージェント設定を保存し、そのプロファイルを、管理対象システムに展開する展開パッケージに適用することができます。また、インストールされたエージェントにポイント設定変更を行い、それらの変更をすぐに適用できます。

設定変更を展開すると、CA Virtual Assurance によってファイル内のテンプレート命令を使用して `sysedge.cf` ファイルがコンパイルされ、CA Virtual Assurance ユーザーインターフェースで入力した情報を適用して設定および自動モニタリングが行われます。コンパイルされた `sysedge.cf` ファイルは、CAM (CA メッセージ) を転送機構として使用して、指定されたシステムに送信されます。CAM は、インストール中に指定されたデータディレクトリにファイルを送信します。SystemEDGE エージェントは、設定ファイルが含まれる CAM メッセージをリスニングして、ファイルが検出されると、ファイルの構文が正しいことを確認してから、自動的にファイルをロードして使用します。CA Virtual Assurance でエージェントを設定すると、エージェント ウォーム スタートがトリガされます。その結果、エージェントに加えた変更が自動的に適用され、再起動する必要がなくなります。

イベント エンジン

CA Virtual Assurance イベント エンジンは、以下のイベントを収集して、SystemEDGE 側および CA Virtual Assurance 側のすべてのアクティビティを追跡します。

- SystemEDGE 状態変更トラップ
- CA Virtual Assurance によって開始された展開および設定変更のステータスを追跡、表示するイベント
- エージェント設定に対する SNMP の変更を CA Virtual Assurance 管理者に通知するイベント

イベント エンジンは、SystemEDGE 例外および操作の追跡と、エージェント設定変更の監査のための信頼できる場所を提供します。

CA Virtual Assurance セキュリティ オプション

CA Virtual Assurance には、エージェントのセキュリティを強化し、他のすべての変更が承認の取得時に使用する必要がある設定の単一ポイントとして CA Virtual Assurance を確立する SystemEDGE セキュリティ ロックダウン オプションが複数用意されています。CA Virtual Assurance によって以下のセキュリティ オプションを設定できます。

SNMP 書き込み削除

SNMP によってエージェントを変更する機能を削除します。この機能は、エージェントを設定する唯一の方法として **CA Virtual Assurance** またはローカル **sysedge.cf** ファイル操作を確立します。**CA Virtual Assurance** には **SNMP** 書き込みアクセスが必要です。

管理モード

CA Virtual Assurance によって開始された変更で、他のメソッドによりエージェントに加えられた変更が上書きされます。**SystemEDGE** のインストール時に **CA Virtual Assurance Manager** ノードに入ると、エージェントは管理モードになります。管理モードにより、他のすべての変更優先する設定のモードとして **CA Virtual Assurance** が確立されます。たとえば、ユーザが直接 **sysedge.cf** ファイルを変更し、その後ファイルが **CA Virtual Assurance** を介してそのシステムに展開された場合、**CA Virtual Assurance** によって配布されたファイルの設定で、ローカルファイル内の設定が上書きされます。

SNMP 変更通知

エージェントが **SNMP Set** 操作によって開始された **SNMP** ベースの変更を適用するとき、**CA Virtual Assurance Manager** に通知します。このオプションは、管理モードのエージェントの場合にのみ選択可能です。デフォルトでは、このオプションはエージェントが管理モードのとき有効です。**CA Virtual Assurance** ユーザ インターフェースから、許容できる変更を判断し、不要な変更はすべて上書きできます。このオプションを有効にすると、**SNMP Set** の変更は、エージェントインストールのデータ ディレクトリ内にある **sysedge_audit.log** ファイルにも記録されます。

これらのセキュリティ オプションの有効化および設定の詳細については、CA Virtual Assurance ドキュメントを参照してください。

AIM アーキテクチャ

Application Insight Module (AIM) は、SystemEDGE エージェント機能を、アプリケーション固有のモニタリングに拡張したり、ベース エージェントに存在しない特殊なモニタリング機能を提供したりします。さまざまなアーキテクチャで多種多様な AIM を利用できますが、以下の規則はすべての AIM に適用されます。

- 通常、AIM はインストール後に自動的にロードされます。AIM インストールプログラムにより、ディレクトリ `SystemEDGE/plugins/plugin-name` にファイル `enable_req` が作成されます。あるいは、エージェントが管理対象外モードの場合、`sysedge.cf` ファイルに `sysedge_plugin <プラグイン名>` パラメータを入力できます。

注: CA Virtual Assurance のインストール時に書き込みアクセスを指定する必要があります。

- SystemEDGE は、ウォーム スタートで AIM をロードできますが、エージェントは以前ロードされた AIM を削除するために完全な再起動が必要です。
- SystemEDGE バージョン 5.0 AIM にはそれぞれ、コア エージェントがサポートする独自の MIB があります。
- 各 AIM には、AIM 設定手順を入力できる独自の設定ファイルを用意できます。

AIM の使用の詳細については、AIM 固有のドキュメントを参照してください。

オンデマンド モジュール

SystemEDGE では、ベース エージェントに以下のオンデマンド モジュールが用意されています。

- Windows のモニタ
- Perfcube

これらのモジュールには、`sysedge.cf` ファイル内の `sysedge_plugin` ステートメントが必要ありません。この機能が必要なとき、SystemEDGE は自動的にオンデマンド モジュールをロードします。

注: 以前トップ プロセス AIM に含まれていたこの機能は、ベース エージェントと統合されました。詳細については、「プロセスとサービスのモニタリング」の章を参照してください。

モジュールおよびモジュールに用意された機能の設定の詳細については、「SystemEDGE AIM」の章を参照してください。

AIM と CA Virtual Assurance の統合

CA Virtual Assurance は、CA Virtual Assurance に同梱されるすべての AIM のリモート展開、および SystemEDGE と SRM AIM のポリシー設定をサポートします。AIM はステートフル モニタを作成することにより、SystemEDGE 状態管理モデルをサポートします。このため、CA Virtual Assurance ユーザ インターフェイスでのレスポンス時間テストに関連付けられたステータス情報を表示できます。

以下の AIM が SystemEDGE リリース 5.7.1 で使用できます。ただし、CA Virtual Assurance での視覚化と設定はサポートされていません。

- IIS AIM
- Apache AIM
- Voice AIM
- CA Insight AIM
- Exchange Server および Active Directory AIM （CA Server Automation には適用できません）

これらの AIM をローカルにロードし、設定できます。しかし、CA Virtual Assurance には、AIM データを解釈するためのプラットフォーム管理モジュールがなく、設定用のインターフェイスも用意されていません。

注: 詳細な AIM サポート マトリックスについては、「SystemEDGE リリース ノート」を参照してください。

SystemEDGE エージェントの使用のガイドライン

システム、ネットワーク、およびアプリケーションの効果的な管理ポリシーをセットアップすることで、SystemEDGE エージェントを最大限に活用できます。具体的には、以下のガイドラインに従います。

- 管理モードのエージェントを **CA Virtual Assurance** から管理して **SNMP** ベースの変更を制限するか、それらの変更をモニタしてエージェント設定を 1 つの一貫したポイントから行えるようにします。
- アクセス制御リストと、プライベート インターフェースへのバインディングによりエージェントへの **SNMP** アクセスを制限します。
 - **SNMP SET** には読み書きコミュニティを使用し、クエリとポーリングには読み取り専用コミュニティを使用します。
 - ポート **161** (**SNMP** 業界標準ポート) または **1691** (SystemEDGE エージェントで使用するために **Internet Assigned Numbers Authority** [IANA] で予約済み) のいずれかを使用します。

注: コミュニティ文字列の指定およびポートの設定の詳細については、「エージェントの設定」の章を参照してください。

- 以下の情報に基づいて、管理ソフトウェアでグループを作成します。
 - オペレーティング システム
 - データベース システム
 - クライアント
 - ファイル サーバ
 - 電子メール サーバ
 - Web サーバ
 - 物理的な場所

注: グループの作成の詳細については、「**CA Virtual Assurance** 管理ガイド」または「**CA eHealth** 管理ガイド」を参照してください。

- 個々のシステムごとではなく、システムのグループに設定ポリシーを定義します。 **CA Virtual Assurance** を使用して、システムのグループに設定セットを展開します。
- **sysedge.cf** ファイルのシステム場所および連絡先フィールド内にわかりやすい情報を含めます。たとえば、ラック **0**、スロット **1**、アトランタ、および電子メール: **it@yourdomain.com** のような情報を含めます。

- モニタリングテーブル全体で行インデックスのよく知られている範囲を使用します。これらのインデックスを定義して行を予約するときは、以下の点に留意してください。
 - 各モニタリングタイプには広い範囲のインデックス番号を使用して、大きくなってもよいようにします。
 - 特定のモニタリングエントリタイプには標準インデックスエントリを使用します。たとえば、利用可能なCPUの合計をモニタするには行10,000を常に使用します。
 - CA Virtual Assuranceを使用して、システムのグループにセルフモニタリングエントリを適用します。

注: 標準インデックス行の使用の詳細については、「セルフモニタリング」の章の「Assigning Entry Rows for the Monitor Table」を参照してください。

- 複数のシステム間で使用可能なエントリを作成するときは、以下の点を考慮します。
 - CPU使用率状態の合計をモニタします（シングルプロセッサシステムおよびマルチプロセッサシステム間で設定を移動できるようになります）。
 - しきい値をモニタし、CA Virtual Assuranceのフラグディングを防ぐために限られた数のトラップ（たとえば、2つか3つ）を送信するようにSystemEDGEを設定します。

注: しきい値を効率的にモニタしてトラップをクリアする方法の詳細については、「セルフモニタリング」の章の「モニタテーブルフラグ」を参照してください。

- 大規模ネットワーク（数百から数千のシステム）を管理していて、詳細なデータを得るために15分未満の間隔で各システムをポーリングしている場合は、以下を実行します。
 - 履歴の収集を使用して、エージェントレベルで非常に詳細なデータを取得し、管理システムで履歴テーブルをポーリングします。短期的な履歴の収集にはSystemEDGEを使用し、長期的な履歴表示にはCA eHealth（または別の管理システム）を使用できます。
 - 詳細については、「履歴の収集」の章を参照してください。
 - エージェントにモニタリングをプッシュし、これらのセルフモニタリングエントリに基づいてトラップを送信するようにエージェントを設定します。

- ステートフル モニタを使用することで、モニタされる 1 つのエントリからの潜在的なトラップの数を制限します。

使用方法のヒントについては、このガイドの以下のセクションを参照してください。

- 「エージェントの設定」の章の「セキュリティの設定に関する推奨事項」
- 「プロセスとサービスのモニタリング」の章の「プロセスとサービスのモニタリングの推奨事項」
- 「ログ ファイル モニタリング」の「ログ ファイル モニタリングの推奨事項」
- 「カスタム MIB オブジェクト」の「Recommendations for Using Extensions」

第 4 章：インストールと展開

この章では、サポートされているすべてのプラットフォーム上で SystemEDGE エージェントのスタンドアロンインストールを実行する方法について説明します。また、CA Virtual Assurance を使用したエージェントの展開についても紹介します。

このセクションには、以下のトピックが含まれています。

[インストールに関する注意事項](#) (P. 92)

[インストール時のエージェントの設定](#) (P. 92)

[CA Virtual Assurance を使用したインストール](#) (P. 93)

[Windows システムでのインストール](#) (P. 93)

[UNIX および Linux システムでのインストール](#) (P. 108)

[応答ファイルの設定と使用](#) (P. 125)

[レガシーモードでのエージェントのインストール](#) (P. 126)

[エージェントの展開](#) (P. 127)

[SystemEDGE のアンインストール](#) (P. 130)

[再インストール](#) (P. 135)

[管理対象ノードおよび AIM サーバのアップグレード](#) (P. 135)

インストールに関する注意事項

SystemEDGE エージェントをインストールする前に、以下のことを考慮してください。

- システムが「SystemEDGE リリース ノート」にリストされているシステム要件を満たしていることを確認します。
- CA_SystemEDGE_Core パッケージによって、SystemEDGE エージェントがインストールされます。
- CA_SystemEDGE_AdvancedEncryption パッケージには、エージェント暗号化ライブラリがあります。Advanced Encryption パッケージのインストールの詳細については、付録「SystemEDGE Advanced Encryption」を参照してください。
- インストール時に定義するほとんどの値は、`CASYSEDGE_DATA¥port<default-port>¥sysedge.cf` ファイル内で変更できます。このファイルには、他の設定オプションもあります。このファイルを使用したエージェントの設定方法の詳細については、「エージェントの設定」の章を参照してください。
- SNMPv1 と SNMPv2c の設定は、インストール時に指定できます。しかし、SNMPv3 の設定は、インストール後に指定する必要があります。詳細については、「エージェントの設定」の章を参照してください。
- インストール操作内で特殊文字または予約文字を使用する場合は、Readme に記載された「*Special Character Limitations*」と、それぞれのオペレーティング環境に基づく制限について考慮してください。オペレーティング環境に基づく制限の例としては、UNIX シェルまたは Windows コマンドプロンプト内の特殊文字や予約文字などがあります。

インストール時のエージェントの設定

SystemEDGE のインストールでは、エージェントをインストールするときに、いくつかの設定タスクを実行できます。たとえば、インストール処理時に、システムの説明と場所、読み取り専用コミュニティや読み書きコミュニティ、SNMPv1 のトラップ先を設定できます。これらの設定は、`sysedge.cf` ファイルを編集することにより、後から変更できます。

注: SNMPv3 ユーザおよびトラップデスティネーションは、インストール中に設定されません。これらの設定は、データディレクトリ内の `sysedgeV3.cf` 設定ファイルを手動で編集して入力する必要があります。SystemEDGE エージェントに SNMPv3 を設定する詳細については、「SNMPv3 設定」を参照してください。

関連項目

[ファイルベースの設定 \(P. 144\)](#)

CA Virtual Assurance を使用したインストール

この章では、SystemEDGE エージェントをスタンドアロンでインストールする方法について説明します。CA Virtual Assurance を使用してエージェントを管理する場合は、CA Virtual Assurance ユーザ インターフェースを使用して、エージェントを展開することをお勧めします。展開の詳細については、「[CA Virtual Assurance によるエージェントの展開 \(P. 128\)](#)」および「CA Virtual Assurance 実装ガイド」を参照してください。

CA Virtual Assurance マネージャのインストールでは、デフォルトでは、SystemEDGE がマネージャ システム上にインストールされます。ローカル システム上での展開はサポートされていません。

Windows システムでのインストール

このセクションでは、Windows システムに SystemEDGE を手動でインストールする方法について説明します。対話型ウィザードを使用してインストールできます。またコマンドラインを使用して無人でインストールすることもできます。サポートされているすべてのハードウェア アーキテクチャごとにパッケージが用意されています。インストーラはハードウェア アーキテクチャを検出し、適切なインストール パッケージを実行します。

以下の状況では、推奨される CA Virtual Assurance 展開方法を使用せずに、エージェントを手動でインストールする必要があります。

- リモート展開をサポートしないシステムにエージェントをインストールする。
- レガシー モードでエージェントをインストールする。

このガイド全体を通じて、**Windows** とは、サポートされているバージョンの Windows を指します。サポートされているバージョンのリストについては、「SystemEDGE リリース ノート」を参照してください。

Windows でのエージェントのインストール

対話型ウィザードを使用して、Windows 用 SystemEDGE エージェントを手動でインストールできます。

次の手順に従ってください：

1. Administrator として Windows システムにログオンし、以下のいずれかを実行します。
 - CA Virtual Assurance インストールイメージの DVD1 で setup.hta をダブルクリックし、[CA Virtual Assurance] ダイアログボックスで [SystemEDGE エージェントのインストール] をクリックします。
 - DVD1¥Installers¥Windows¥Agent¥SysMan¥CA_SystemEDGE_Core フォルダを開き、ca-setup.exe をダブルクリックします。

注：Windows Vista 以降を実行するシステムでは、非管理者としてインストールできます。オペレーティングシステムによって、管理者の認証情報を使用してインストールを認可するように要求されます。

インストール ウィザードが起動します。インストーラがシステムのハードウェアアーキテクチャを検出し、適切なバージョンのインストールパッケージを実行します。

2. [次へ] をクリックします。
[使用許諾契約] ページが表示されます。
3. 使用許諾契約を読み、下部までスクロールします。[使用許諾契約書に同意します] を選択し、[次へ] をクリックします。
[インストール タイプ] ページが表示されます。
4. [標準] または [カスタム] を選択し、[次へ] をクリックします。

注：以下のプロシージャではカスタム インストールについて説明します。[標準] を選択して [次へ] をクリックすると、[設定の確認] ページが表示されます。

[デスティネーション場所] ダイアログ ボックスが表示されます。

5. インストール場所およびデータ ディレクトリとしてデフォルトを受け入れるか、参照して選択し、[次へ] をクリックします。手順 9 に進みます。別の場所を指定する場合は、[詳細] をクリックし、手順 8 に進みます。

デスティネーション場所

エージェントをインストールする場所を指定します。デフォルトでは、インストール ディレクトリは *Install_Path\SystemEDGE* です。また、ランタイム プログラム データは *config* サブディレクトリに保存されます。その他のパラメータを指定するには、[詳細] をクリックします。

注: 以前のバージョンのエージェントがすでにインストールされているシステムにエージェントをインストールする場合、インストーラは既存のエージェントのインストール ディレクトリを選択します。

[詳細] ダイアログ ボックスを省略すると、[構成マネージャ設定] ダイアログ ボックスが表示されます。

6. [詳細なデスティネーション場所] ダイアログ ボックスで以下のフィールドに入力し、[次へ] をクリックします。

SystemEDGE バイナリ パス

プログラム バイナリおよびドキュメント用のディレクトリを指定します。

SystemEDGE データ パス

ランタイム プログラム データ用のディレクトリを指定します。

CA 共有コンポーネント パス

CA 共有コンポーネント用のディレクトリを指定します。いずれかの CA ソフトウェアによって設定されると、このディレクトリは変更できません。また、ユーザ インターフェース内の対応するフィールドは無効になります。

[構成マネージャ設定] ダイアログ ボックスが表示されます。

7. 次のフィールドの入力を完了し、[次へ] をクリックします。

構成マネージャ ホスト名

このエージェントを管理する構成マネージャのホスト名を指定します。**CA Virtual Assurance** が実行されるシステムからこのエージェントを設定できるようにするために、このパラメータの値を入力します。アスタリスク (*) を入力すると、エージェントシステムを最初に検出したマネージャが、このパラメータに設定されます。

デフォルトの設定ポリシー名

エージェントが使用する、**CA Virtual Assurance** マネージャによって保持される設定ポリシー ファイルの名前を指定します。このパラメータの値を入力して、マネージャからの既存の設定ファイルに従って **SystemEDGE** を設定します。

システム上でネイティブ Microsoft SNMP エージェントが実行されていることをインストーラが検知すると、以下の [ネイティブ SNMP エージェント オプション] ダイアログ ボックスが表示されます。

8. SNMP (Simple Network Management Protocol) がシステムにインストールされている場合は、以下のいずれかのオプションを選択して、[次へ] をクリックします。

既存の SNMP エージェントのデフォルト

ネイティブ **SNMP** エージェントから継承されるデフォルトの設定を使用するかどうかを指定します。ネイティブ **SNMP** エージェントとは異なるコミュニティ文字列およびトラップ先を使用する場合は、このチェック ボックスをオフにしておきます。

ネイティブ SNMP エージェントの無効化

ネイティブ **SNMP** エージェントを停止して無効にするかどうかを指定します。ネイティブ **SNMP** エージェントを有効にしておく場合は、別のポート上で **SystemEDGE** を実行します。

9. 次のフィールドの入力を完了し、[次へ] をクリックします。

SNMP のポート番号

SystemEDGE エージェントを実行するポートを指定します。

デフォルト : **161**

重要: このポート番号は、**SystemEDGE** エージェント専用になります。ほかのアプリケーションがこのポート番号を使用している場合、インストールは失敗します。ネイティブ **SNMP** エージェントがすでにデフォルトポートを使用している場合は、たとえば **1691** や **6665** など、別のポートを指定します。

[SNMP システム情報] ダイアログ ボックスが表示されます。

10. 次のフィールドの入力を完了し、[次へ] をクリックします。

システムの説明

システムに関する情報（システム名など）を指定します。この情報は、**sysDescr MIB-II** オブジェクトに入力されます。

システムの場所

システムの場所を指定します。この値は、**sysLocation MIB-II** オブジェクトに入力されます。

システム担当者

システム担当者の情報を指定します。この情報は、**sysContact MIB-II** オブジェクトに入力されます。

[SNMP コミュニティ設定] ダイアログ ボックスが表示されます。

11. 以下のフィールドに入力し、[次へ] をクリックします。手順 13 に進みます。複数のコミュニティ文字列を指定する場合は、[詳細] をクリックして、手順 12 に進みます。

読み取り専用コミュニティ

SNMP 読み取り専用コミュニティ文字列を指定します。

デフォルト : **public**

読み取り/書き込みコミュニティ

SNMP 読み取り/書き込みコミュニティ文字列を指定します。

[詳細] ダイアログ ボックスを省略すると、[SNMP トラップ設定] ダイアログ ボックスが表示されます。

12. [SNMP コミュニティ設定 - 詳細] ダイアログ ボックスで以下のフィールドに入力し、[次へ] をクリックします。

読み取り専用コミュニティ

SNMP 読み取り専用コミュニティ文字列を指定します。複数のコミュニティを指定する場合は、セミコロンで区切ります（たとえば、**public1; public2**）。また、アクセスを制限するために、各コミュニティに対して IP アドレスのリストを含めることができます（たとえば、**public 1.2.3.4**）。

デフォルト： **public**

読み取り/書き込みコミュニティ

SNMP 読み取り/書き込みコミュニティ文字列を指定します。複数のコミュニティを指定する場合は、セミコロンで区切ります（たとえば、**rwcomm1; rwcomm2**）。また、アクセスを制限するために、各コミュニティに対してスペースで区切られた IP アドレスのリストを含めることができます（たとえば、**rwcomm1 1.2.3.4**）。読み取り/書き込みコミュニティは、いくつかの AIM（たとえば、RM）の正しい操作、およびいくつかのリモート用途（たとえば、モニタの作成）を行うために必要です。

[SNMP トラップ設定] ダイアログ ボックスが表示されます。

13. 以下のフィールドに入力し、[次へ] をクリックします。手順 15 に進みます。複数のトラップ先を指定する場合は、[詳細] をクリックして、手順 14 に進みます。

トラップ コミュニティ文字列

送信されたトラップ メッセージ内のエンコードされた SNMP コミュニティを指定します。

デフォルト： **public**

デスティネーション ホスト

トラップ メッセージのデスティネーションを指定します。

デフォルト： [構成マネージャ設定] ダイアログ ボックスで設定される **構成マネージャ ホスト名**。

ポート番号

トラップ メッセージが送信されるポートを指定します。

デフォルト： **162**

[詳細] ダイアログ ボックスを省略すると、[その他の設定] ダイアログ ボックスが表示されます。

14. [SNMP トラップ設定 - 詳細] ダイアログ ボックスで以下のフィールドに入力し、[次へ] をクリックします。

トラップ設定

1 つ以上のトラップ先を指定します。複数のエントリをセミコロンの区切って指定できます (たとえば、**public server1; public server2 1162**)。

[その他の設定] ダイアログ ボックスが表示されます。

15. 以下のフィールドに入力し、[次へ] をクリックします。

インストール後に開始

エージェントがインストールの最後に開始されるかどうかを指定します。

インストールドキュメント

ドキュメントをインストールするかどうかを指定します。

[設定の確認] ページが表示されます。

16. インストール設定を確認し、[インストール] をクリックします。
インストールが終了すると、[インストールが完了しました] ページが表示されます。
17. [終了] をクリックします。
インストールは終了です。

コマンドラインを使用した Windows へのエージェントのインストール

コマンドラインバージョンのインストーラを使用して、SystemEDGE Windows パッケージをインストールできます。コマンドラインからインストールするときは、パラメータを使用して各種のインストールプロパティを設定します。コマンドラインから以下を行うことができます。

- インストールパラメータを事前に入力して（または入力せずに）インストールウィザードを開始する
- インストールパラメータを指定して、対話式ウィザードなしでインストーラを実行する

以下の手順では、この 2 番目のシナリオ（コマンドラインからの無人インストールの実行）について説明します。

次の手順に従ってください：

1. Windows システムに管理者としてログオンします。

注：Windows Vista 以降を実行しているシステムでは、管理者以外の権限でもインストールできます。その場合、管理者の認証情報でインストールを許可するようにオペレーティングシステムから促されます。

2. コマンドプロンプトを開き、
DVD1¥Installers¥Windows¥Agent¥SysMan¥CA_SystemEDGE_Core フォルダに移動して、以下の必須パラメータを入力します（手順 3 を完了するまで、コマンドを実行しないでください）。

```
ca-setup CA_SETUP_MODE=UNATTENDED EULA_ACCEPTED="YES" [parameter]
```

注：インストールパラメータのヘルプを参照するには、ca-setup -? を コマンドプロンプトで入力してください。

CA_SETUP_MODE

インストールモードを指定します。サイレントモードでインストールを実行するには、このパラメータを **UNATTENDED** に設定します。このパラメータを省略すると、コマンド実行後に、インストールウィザードはあらかじめ指定されたパラメータ値を使用して開始されます。

EULA_ACCEPTED

使用許諾契約を読み取り、使用許諾契約に同意するかどうかを指定します。インストールモードを **UNATTENDED** に設定している場合に、このパラメータを省略するか、YES 以外に設定すると、インストールは失敗します。このパラメータは、対話式インストールでは不要です。

- 必要に応じてオプションのパラメータを追加し、コマンドを実行します。

注: 以下のオプションのパラメータを省略する場合は、値は必要ありません。

CA_SETUP_LOG_FILE
CA_SETUP_VERBOSE
CASE_INSTALLDIR
CASE_PUBDATAIDIR
CASE_SNMP_PORT
CASE_SNMP_SYS_DESC
CASE_SNMP_SYS_LOC
CASE_SNMP_SYS_CONTACT
CASE_SNMP_READ_COMMUNITY
CASE_SNMP_READ_ALLOWED_MANAGERS
CASE_SNMP_WRITE_COMMUNITY
CASE_SNMP_WRITE_ALLOWED_MANAGERS
CASE_SNMP_TRAP_COMMUNITY
CASE_SNMP_TRAP_DESTINATION
CASE_SNMP_TRAP_PORT
CASE_DISABLE_NATIVE_SNMP
CASE_DEFAULT_FROM_NATIVE_SNMP
CASE_MANAGER_HOSTNAME
CASE_MANAGER_POLICY_NAME
CASE_START_AFTER_INSTALL
CASE_INSTALL_DOCS
CASE_LEGACY_MODE

CA_SETUP_LOG_FILE

インストールメッセージをログ記録する場所およびファイル名を指定します。

CA_SETUP_VERBOSE

「yes」に設定すると、インストーラが詳細モードになります。詳細モードでは、インストールログファイルにより詳細な情報がログ記録されます。

CASE_INSTALLDIR

SystemEDGE のインストールディレクトリを指定します。このディレクトリには、**AIM**、**Advanced Encryption** など、**SystemEDGE** 関連のものすべてが含まれており、コア インストーラによって設定された後は、変更されることはありません。

注: デフォルト以外のインストールディレクトリを定義すると、インストーラは、**SystemEDGE** サブフォルダを作成せずに、指定されたディレクトリにエージェント ファイルを直接インストールします。

デフォルト : **C:\Program Files\CA\SystemEDGE**

CASE_PUBDATADIR

SystemEDGE データ ディレクトリを指定します。すべての設定は、このディレクトリで実行され、動的なデータが保存されます（このドキュメント内では、**SystemEDGE** データ ディレクトリを変数 **CASYSEDGE_DATA** と呼んでいます）。エージェントの設定ファイルは、**SNMP_PORT** パラメータの値に基づいて、ポートに固有のサブディレクトリ内に配置されます。

デフォルト : **C:\Documents and Settings\All Users\Application Data\CA\SystemEDGE** (Windows XP および 2003) 、
C:\Users\Public\CA\SystemEDGE (Windows Vista および 2008)

CASE_SNMP_PORT

SystemEDGE によって使用されるポートを指定します。この値は、**CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承することもできます。

CASE_SNMP_PORT パラメータを使用してポートを指定すると、継承された値が上書きされます。このポートは一意である必要があります。一意でないと、インストールは失敗します。デフォルトポートである **161** がネイティブ **SNMP** エージェントによってすでに使用されている場合（そして、このエージェントを無効にする計画がない場合）、たとえば **1691** や **6665** など、別の一意のポートを指定する必要があります。

デフォルト : **161**

CASE_SNMP_SYS_DESC

sysDescr MIB-II オブジェクトに入力されるシステムについての情報（システム名など）を指定します。この値も

CASE_DEFAULT_FROM_NATIVE_SNMP パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。

CASE_SNMP_SYS_DESC パラメータを使用して説明を指定した場合は、継承された値が上書きされます。

CASE_SNMP_SYS_LOC

sysLocation MIB-II オブジェクトに入力されるシステムの場所を指定します。この値も **CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。

CASE_SNMP_SYS_LOC パラメータを使用して場所を指定した場合は、継承された値が上書きされます。

CASE_SNMP_SYS_CONTACT

sysContact MIB-II オブジェクトに入力されるシステム担当者情報を指定します。この値も **CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。 **CASE_SNMP_SYS_CONTACT** パラメータを使用して担当者を指定した場合は、継承された値が上書きされます。

CASE_SNMP_READ_COMMUNITY

エージェントへ **GET** 要求を送信できる **SNMP** 読み取りコミュニティの名前を指定します。セミコロンで区切るにより、複数のコミュニティを指定できます (たとえば **public1; public2**)。また、コミュニティごとにスペースで区切った IP アドレス リストを含めて、アクセスを制限できます (たとえば **public 1.2.3.4**)。この値も **CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。

CASE_SNMP_READ_COMMUNITY パラメータを使用して読み取りコミュニティを指定した場合は、継承された値が上書きされます。

デフォルト: **snmp_public** (新規インストールで (アップグレードは除く)、読み取り/書き込みコミュニティが指定されていない場合のみ有効)

CASE_SNMP_READ_ALLOWED_MANAGERS

SNMP_READ_COMMUNITY を持つエージェントにクエリすることを許された **SNMP** マネージャの IP アドレスまたはホスト名をスペースで区切って指定します。リストを指定する場合、**SNMP_READ_COMMUNITY** には、1 つの語 (**SNMP** コミュニティ) のみが含まれる必要があります。

CASE_SNMP_WRITE_COMMUNITY

エージェントへ **GET** 要求と **SET** 要求を送信できる **SNMP** 書き込みコミュニティの名前を指定します。セミコロンで区切るにより、複数のコミュニティを指定できます (たとえば **rwcomm1;rwcomm2**)。また、コミュニティごとにスペースで区切った IP アドレス リストを含めて、アクセスを制限できます (たとえば **rwcomm1 1.2.3.4**)。この値も

CASE_DEFAULT_FROM_NATIVE_SNMP パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。

CASE_SNMP_WRITE_COMMUNITY パラメータを使用して書き込みコミュニティを指定した場合は、継承された値が上書きされます。

CASE_SNMP_WRITE_ALLOWED_MANAGERS

SNMP_WRITE_COMMUNITY を持つエージェントにクエリすることを許された **SNMP** マネージャの IP アドレスまたはホスト名をスペースで区切って指定します。リストを指定する場合、**SNMP_WRITE_COMMUNITY** には、1 つの語 (**SNMP** コミュニティ) のみが含まれる必要があります。

CASE_SNMP_TRAP_COMMUNITY

SNMP トラップ コミュニティおよびトラップ先アドレスを指定します。セミコロン (;) で区切るにより複数のトラップ コミュニティ設定を指定できます。このパラメータの値は、**CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承することもできます。**CASE_SNMP_TRAP_COMMUNITY** パラメータを使用してトラップ コミュニティを指定すると、継承された値が上書きされます。以下の値がこのパラメータに必要です。

- コミュニティ名
- トラップの送信先のデスティネーションアドレス

以下の値は、このパラメータのオプションです。

- トラップの送信先のポート番号
- トラップ ソースのエンコーディング オプション
- トラップ ソースのホスト名

トラップ コミュニティ設定の構文：

```
community-string {ip-address|hostname} [port [encoding [source]]]
```

例：

```
public 1.2.3.4;public 2.3.4.5 1162;trapcom 3.4.5.6 1162 100 4.5.6.7
```

CASE_SNMP_TRAP_DESTINATION

トラップの送信先のホスト名または IP アドレスを指定します。指定する場合、**SNMP_TRAP_COMMUNITY** には、1 つの語 (**SNMP** コミュニティ) のみが含まれる必要があります、また **SNMP_TRAP_PORT** を指定する必要があります。

CASE_SNMP_TRAP_PORT

トラップの送信先のデスティネーション ポート番号を指定します。指定する場合、**SNMP_TRAP_COMMUNITY** には、1 つの語 (**SNMP** コミュニティ) のみが含まれる必要があります、また **SNMP_TRAP_DESTINATION** を指定する必要があります。

CASE_DISABLE_NATIVE_SNMP

ネイティブ **SNMP** エージェントを停止して無効にするかどうかを指定します。

デフォルト : **no**

CASE_DEFAULT_FROM_NATIVE_SNMP

ネイティブ **SNMP** エージェントのデフォルトの **SNMP** 設定を使用するかどうかを指定します。

デフォルト : **no**

CASE_MANAGER_HOSTNAME

このエージェントを管理する構成マネージャのホスト名を指定します。このパラメータに値を入力すると、指定したマネージャからこのエージェントを設定できます。アスタリスク (*) を入力すると、エージェント システムを最初に検出したマネージャが受け入れられます。このマネージャは、エージェントの設定をフル コントロールできるようになります。デフォルトでは、マネージャホストは入力（使用）されず、エージェントは管理対象外モードで実行されます。

CASE_MANAGER_POLICY_NAME

エージェントで使用する必要がある、構成マネージャのポリシーファイルの名前を指定します。このパラメータに値を入力すると、マネージャの既存の設定ファイルに従って **SystemEDGE** が設定されます。デフォルトでは、エージェントはインストール済みのポリシー ファイルを使用します。

CASE_START_AFTER_INSTALL

インストールの完了後にエージェントを自動的に開始するかどうかを指定します。

デフォルト : **yes**

CASE_INSTALL_DOCS

SystemEDGE のドキュメントをエージェントと同時にインストールするかどうかを指定します。

デフォルト : **yes**

CASE_LEGACY_MODE

エージェントをレガシー モードでインストールするかどうかを指定します。レガシー モードでは、ベース エージェントのみがインストールされ、**CA Virtual Assurance** でのエージェントの使用を円滑にする要素は、まったくインストールされません。 **CA Virtual Assurance** でエージェントを使用しない場合は、レガシー モードを使用してインストールしてください。

エージェントを管理対象モードに変更するには、エージェントを再インストールまたはアップグレードし、**CASE_LEGACY_MODE=no** と指定します。

デフォルト : **no**

コマンドラインで、コマンドおよびすべての必須パラメータおよび値を入力します。Enter キーを押してインストールを開始します。 インストーラは、オペレーティング システムのハードウェア アーキテクチャを検出し、インストーラの適切なバージョンを実行します。

注: ユーザがインストールに関する使用許諾に同意しない場合、インストールは失敗します。

インストールを確認するには、[Windows サービス] ダイアログ ボックスに **CA SystemEDGE** サービスがあるかどうかを確認します。または、インストール ディレクトリ内に **SystemEDGE** ファイル、あるいは [プログラムの追加と削除] ダイアログ ボックス内に **CA SystemEDGE** コアがあるかどうかを確認します。 インストール ログ ファイルを指定した場合は、そのファイルをチェックしてインストールの成功を確認することもできます。

注: Windows Vista 以降を実行するシステムにインストールする場合は、**SystemEDGE** インストーラが「Microsoft Visual C++ 2005 再頒布可能パッケージ」を自動的にインストールします。 このパッケージがインストールされていないと、**SystemEDGE** は機能しません。「Microsoft Visual C++ 2005 再頒布可能パッケージ」をインストールするには、表示された使用許諾契約に同意する必要があります。「Microsoft Visual C++ 2005 再頒布可能パッケージ」の使用許諾契約は、対話式的 **SystemEDGE** インストーラには表示されません。

UNIX および Linux システムでのインストール

ここでは、UNIX および LINUX のシステム上で SystemEDGE エージェントを手動でインストールする方法について説明します。対話型ウィザードを使用してインストールできます。またコマンドラインを使用して無人でインストールすることもできます。対話型ウィザードはテキストモードでコンソール上に表示されます。Xserver が利用でき、DISPLAY 環境が正しく設定されている場合は、グラフィカルなアプリケーションとして表示されます。

以下の状況では、推奨される CA Virtual Assurance 展開方法を使用せずに、エージェントを手動でインストールする必要があります。

- リモート展開をサポートしないシステムにエージェントをインストールする。
- レガシーモードでエージェントをインストールする。

サポートされる UNIX および LINUX のプラットフォームおよびバージョンの詳細については、「[SystemEDGE リリースノート](#)」を参照してください。

関連項目

[レガシーモードでのエージェントのインストール \(P. 126\)](#)

UNIX システムおよび LINUX システムでのエージェントのインストール

対話式ウィザードを使用して、UNIX および Linux に手動で SystemEDGE エージェントをインストールできます。

注:

- このドキュメントでは、インストールディレクトリを CASYSEGE と表し、データディレクトリを CASYSEGE_DATA と表します。
- インストールプログラムは、/etc/profile 内のシステム環境設定を変更しません。

次の手順に従ってください:

1. root ユーザとしてシステムにログインし、DVD2 をマウントします。
2. 端末コンソールを開き、`Installers/platform/Agent/SysMan/CA_SystemEDGE_Core` ディレクトリに移動します (使用中のオペレーティング システムに対応する *platform* ディレクトリを選択します)。
3. このディレクトリから、以下のようにインストーラを実行します。

```
sh ca-setup.sh
```

インストーラの [はじめに] ページが表示されます。

4. [次へ] をクリックします。
[使用許諾契約書] ページが表示されます。
5. 使用許諾契約を読み、[使用許諾契約書に同意します] を選択します。 [次へ] をクリックします。
[インストールタイプ] ページが表示されます。
6. [標準] または [カスタム] を選択し、[次へ] をクリックします。

注: 以下の手順では、カスタム インストールについて説明します。 [標準] を選択した場合は、[次へ] をクリックすると [設定の確認] ページが表示されます。

[デスティネーション場所] ダイアログ ボックスが表示されます。

7. インストール場所およびデータ ディレクトリとしてデフォルトを受け入れるか、参照して選択し、[次へ] をクリックします。手順 9 に進みます。別の場所を指定する場合は、[詳細] をクリックして手順 8 に進みます。

デスティネーション場所

エージェントをインストールする場所を指定します。デフォルトでは、インストールディレクトリは `/opt/CA/SystemEDGE` であり、ランタイムプログラム データは `config` サブディレクトリに保存されます。その他のパラメータを指定するには、[詳細] をクリックします。

注: 旧バージョンのエージェントがすでにインストールされているシステムにエージェントをインストールする場合は、インストーラが既存のエージェントのインストールディレクトリを自動的に選択します。

[詳細] ダイアログ ボックスを省略すると、[構成マネージャ設定] ダイアログ ボックスが表示されます。

8. [詳細なデスティネーション場所] ダイアログ ボックスで以下のフィールドに入力し、[次へ] をクリックします。

SystemEDGE バイナリ パス

プログラム バイナリおよびドキュメント用のディレクトリを指定します。

SystemEDGE データ パス

ランタイム プログラム データ用のディレクトリを指定します。

CA 共有コンポーネント パス

CA 共有コンポーネント用のディレクトリを指定します。いずれかの CA ソフトウェアによって設定されると、このディレクトリは変更できません。また、ユーザ インターフェース内の対応するフィールドは無効になります。

[構成マネージャ設定] ダイアログ ボックスが表示されます。

9. 次のフィールドの入力を完了し、[次へ] をクリックします。

構成マネージャ ホスト名

このエージェントを管理する構成マネージャのホスト名を指定します。CA Virtual Assurance が実行されるシステムからこのエージェントを設定できるようにするために、このパラメータの値を入力します。アスタリスク (*) を入力すると、エージェントシステムを最初に検出したマネージャが、このパラメータに設定されます。

デフォルトの設定ポリシー名

エージェントが使用するポリシー ファイル (CA Virtual Assurance マネージャによって管理されるファイル) の名前を指定します。このパラメータの値を入力して、マネージャからの既存の設定ファイルに従って SystemEDGE を設定します。

インストーラがシステム上で実行されているネイティブ SNMP エージェントを検出すると、[ネイティブ SNMP エージェント オプション] ダイアログ ボックスが表示されます。

10. 次のフィールドの入力を完了し、[次へ] をクリックします。

既存の SNMP エージェントのデフォルト

ネイティブ **SNMP** エージェントから継承されるデフォルトの設定を使用するかどうかを指定します。ネイティブ **SNMP** エージェントとは異なるコミュニティ文字列およびトラップ先を使用する場合は、このチェック ボックスをオフにしておきます。

ネイティブ SNMP エージェントの無効化

ネイティブ **SNMP** エージェントを停止して無効にするかどうかを指定します。ネイティブ **SNMP** エージェントを有効にしておく場合は、別のポート上で **SystemEDGE** を実行します。

11. 次のフィールドの入力を完了し、[次へ] をクリックします。

SNMP のポート番号

SystemEDGE エージェントを実行するポートを指定します。他のアプリケーションが使用していないポートを指定してください。他のアプリケーションがこのポートを使用していると、インストールは失敗します。ネイティブ **SNMP** エージェントがすでにデフォルトポートを使用している場合は、たとえば **1691** や **6665** など、別のポートを指定します。

デフォルト : **161**

[SNMP システム情報] ダイアログ ボックスが表示されます。

12. 次のフィールドの入力を完了し、[次へ] をクリックします。

システムの説明

システムに関する情報 (システム名など) を指定します。この情報は、**sysDescr MIB-II** オブジェクトに入力されます。

システムの場所

システムの場所を指定します。この値は、**sysLocation MIB-II** オブジェクトに入力されます。

システム担当者

システム担当者の情報を指定します。この情報は、**sysContact MIB-II** オブジェクトに入力されます。

[SNMP コミュニティ設定] ダイアログ ボックスが表示されます。

13. 以下のフィールドに入力し、[次へ]をクリックします。手順 15 に進みます。複数のコミュニティ文字列を指定する場合は、[詳細] をクリックして手順 14 に進みます。

読み取り専用コミュニティ

SNMP 読み取り専用コミュニティ文字列を指定します。

デフォルト : **public**

読み取り/書き込みコミュニティ

SNMP 読み取り/書き込みコミュニティ文字列を指定します。

[詳細] ダイアログ ボックスを省略すると、[SNMP トラップ設定] ダイアログ ボックスが表示されます。

14. [SNMP コミュニティ設定 - 詳細]ダイアログ ボックスで以下のフィールドに入力し、[次へ] をクリックします。

読み取り専用コミュニティ

SNMP 読み取り専用コミュニティ文字列を指定します。 個々のコミュニティをセミコロンで区切ることによって、複数のコミュニティを指定できます (たとえば、**public1;public2**)。また、アクセスを制限するために、各コミュニティに対して IP アドレスのリストを含めることができます (たとえば、**public 1.2.3.4**)。

デフォルト : **public**

読み取り/書き込みコミュニティ

SNMP 読み取り/書き込みコミュニティ文字列を指定します。 個々のコミュニティをセミコロンで区切ることによって、複数のコミュニティを指定できます (たとえば、**rwcomm1;rwcomm2**)。また、アクセスを制限するために、各コミュニティに対してスペースで区切られた IP アドレスのリストを含めることができます (たとえば、**rwcomm1 1.2.3.4**)。読み取り/書き込みコミュニティは、いくつかの AIM (たとえば、RM) の正しい操作、およびいくつかのリモート用途 (たとえば、モニタの作成) を行うために必要です。

[SNMP トラップ設定] ダイアログ ボックスが表示されます。

15. 以下のフィールドに入力し、[次へ] をクリックして手順 17 に進みます。複数のトラップ デスティネーションを指定する場合は、[詳細] をクリックして手順 16 に進みます。

トラップ コミュニティ文字列

送信されたトラップ メッセージ内のエンコードされた **SNMP** コミュニティを指定します。

デフォルト: **public**

デスティネーション ホスト

トラップ メッセージのデスティネーションを指定します。

デフォルト: [構成マネージャ設定] ダイアログ ボックスで設定される **構成マネージャ** ホスト名。

ポート番号

トラップ メッセージが送信されるポートを指定します。

デフォルト: **162**

[詳細] ダイアログ ボックスをスキップすると、[権限分離ユーザ] ダイアログ ボックスが表示されます。

16. [SNMP トラップ設定 - 詳細] ダイアログ ボックスで以下のフィールドに入力し、[次へ] をクリックします。

トラップ設定

1 つ以上のトラップ先を指定します。複数のエントリをセミコロンで区切って指定できます (たとえば、**public server1;public server2 1162**) 。

[権限分離ユーザ] ダイアログ ボックスが表示されます。

17. 以下のフィールドに入力して、[次へ] をクリックします。

ユーザ名

SNMP 通信中にエージェントが使用して実行する認証情報のユーザ名を指定します。

このエントリは、別のユーザ アカウントで **SNMP** 通信を実行するようにエージェントに指示します (**UNIX** のみ) 。このエージェントはまた、有効なグループとして、このユーザのデフォルト グループを使用します。

デフォルト: エージェントは **root** アカウントを使用して動作します。

[その他の設定] ダイアログ ボックスが表示されます。

18. 以下のフィールドに入力し、[次へ] をクリックします。

インストール後に開始

エージェントがインストールの最後に開始されるかどうかを指定します。

インストールドキュメント

ドキュメントをインストールするかどうかを指定します。

[設定の確認] ページが表示されます。

19. インストール設定を確認し、[インストール] をクリックします。

インストールが終了すると、[インストールが完了しました] ページが表示されます。

20. [終了] をクリックします。

インストールは終了です。

64 ビット版 Linux のリリースでの SystemEDGE インストールが失敗する

症状:

64 ビット版 Linux のリリースに SystemEDGE をインストールすると、インストールが失敗します。

解決方法:

64 ビット版 Linux のリリースに SystemEDGE を実行しインストールするには、必要な 32 ビット ライブラリをインストールします。

- Red Hat または SuSE のディストリビューション上で有効:

```
yum install glibc.i686
```

- Debian ディストリビューション上で有効:

```
apt-get install ia32-libs
```

コマンドラインを使用した UNIX へのエージェントのインストール

インストーラのコマンドラインバージョンを使用して **SystemEDGE UNIX** パッケージをインストールできます。コマンドラインからインストールするときは、パラメータを使用して各種のインストールプロパティを設定します。コマンドラインから以下を行うことができます。

- インストールパラメータを事前に入力して（または入力せずに）インストーラウィザードを開始する
- インストールパラメータを指定して、対話式ウィザードなしでインストーラを実行する

注:

- UNIX では、応答ファイルの自動作成によるサイレントインストールの実行はサポートされません。ただし、応答ファイルを手動で作成して無人インストールに使用することはできます。
- インストールプログラムは、**/etc/profile** 内のシステム環境設定を変更しません。

以下の手順では、この 2 番目のシナリオ（コマンドラインからの無人インストールの実行）について説明します。

次の手順に従ってください：

1. システムに **root** としてログオンします。
2. **DVD2/Installers/platform/Agent/SysMan/CA_SystemEDGE_Core** ディレクトリに移動します（使用中のオペレーティングシステムに対応する **platform** ディレクトリを選択します）。次に、以下の必須パラメータを入力します（手順 3 を完了するまでコマンドを**実行しない**でください）。

```
sh ca-setup.sh CA_SETUP_MODE="UNATTENDED" EULA_ACCEPTED="yes" [parameter]
```

注：インストールパラメータのヘルプを参照するには、**ca-setup -?** を コマンドプロンプトで入力してください。

CA_SETUP_MODE

インストールモードを指定します。インストールウィザードを表示せずにインストールを実行するには、このパラメータを **UNATTENDED** に設定します。このパラメータを省略すると、コマンド実行後に、インストールウィザードはあらかじめ指定されたパラメータ値を使用して開始されます。

EULA_ACCEPTED

使用許諾契約を読み取り、使用許諾契約に同意するかどうかを指定します。インストールモードを **UNATTENDED** に設定している場合に、このパラメータを省略するか、YES 以外に設定すると、インストールは失敗します。このパラメータは、対話式インストールでは不要です。

- 必要に応じてオプションのパラメータを追加し、コマンドを実行します。

注: 以下のオプションのパラメータを省略する場合は、値は必要ありません。

CA_SETUP_LOG_FILE
CA_SETUP_VERBOSE
CASE_INSTALLDIR
CASE_PUBDATA DIR
CASE_SNMP_PORT
CASE_SNMP_SYS_DESC
CASE_SNMP_SYS_LOC
CASE_SNMP_SYS_CONTACT
CASE_SNMP_READ_COMMUNITY
CASE_SNMP_READ_ALLOWED_MANAGERS
CASE_SNMP_WRITE_COMMUNITY
CASE_SNMP_WRITE_ALLOWED_MANAGERS
CASE_SNMP_TRAP_COMMUNITY
CASE_SNMP_TRAP_DESTINATION
CASE_SNMP_TRAP_PORT
CASE_DISABLE_NATIVE_SNMP
CASE_DEFAULT_FROM_NATIVE_SNMP
CASE_MANAGER_HOSTNAME
CASE_MANAGER_POLICY_NAME
CASE_PRIVSEP_USER
CASE_START_AFTER_INSTALL
CASE_INSTALL_DOCS
CASE_LEGACY_MODE

CA_SETUP_LOG_FILE

インストールメッセージをログ記録する場所およびファイル名を指定します。デフォルトでは、
/opt/CA/installer/log/CA_SETUP_PACKAGE_NAME.log にメッセージが記録されます。

CA_SETUP_VERBOSE

「yes」に設定すると、インストーラが詳細モードになります。詳細モードでは、インストールログファイルにより詳細な情報がログ記録されます。

CASE_INSTALLDIR

SystemEDGE のインストールディレクトリを指定します。このディレクトリには、**AIM**、**Advanced Encryption** など、**SystemEDGE** 関連のものすべてが含まれており、コア インストーラによって設定された後は、変更されることはありません。

注: デフォルト以外のインストールディレクトリを定義すると、インストーラは、**SystemEDGE** サブフォルダを作成せずに、指定されたディレクトリにエージェント ファイルを直接インストールします。

デフォルト : **/opt/CA/SystemEDGE**

CASE_PUBDATADIR

SystemEDGE データ ディレクトリを指定します。すべての設定は、このディレクトリで実行され、動的なデータが保存されます（このドキュメント内では、**SystemEDGE** データ ディレクトリを変数 **CASYSEDGE_DATA** と呼んでいます）。エージェントの設定ファイルは、**SNMP_PORT** パラメータの値に基づいて、ポートに固有のサブディレクトリ内に配置されます。

デフォルト : **/opt/CA/SystemEDGE/config**

CASE_SNMP_PORT

SystemEDGE によって使用されるポートを指定します。この値は、**CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承することもできます。

CASE_SNMP_PORT パラメータを使用してポートを指定すると、継承された値が上書きされます。このポートは一意である必要があります。一意でないと、インストールは失敗します。デフォルトポートである **161** がネイティブ **SNMP** エージェントによってすでに使用されている場合（そして、このエージェントを無効にする計画がない場合）、たとえば **1691** や **6665** など、別の一意のポートを指定する必要があります。

デフォルト : **161**

CASE_SNMP_SYS_DESC

sysDescr MIB-II オブジェクトに入力されるシステムについての情報（システム名など）を指定します。この値も

CASE_DEFAULT_FROM_NATIVE_SNMP パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。

CASE_SNMP_SYS_DESC パラメータを使用して説明を指定した場合は、継承された値が上書きされます。

CASE_SNMP_SYS_LOC

sysLocation MIB-II オブジェクトに入力されるシステムの場所を指定します。この値も **CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。

CASE_SNMP_SYS_LOC パラメータを使用して場所を指定した場合は、継承された値が上書きされます。

CASE_SNMP_SYS_CONTACT

sysContact MIB-II オブジェクトに入力されるシステム担当者情報を指定します。この値も **CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。**CASE_SNMP_SYS_CONTACT** パラメータを使用して担当者を指定した場合は、継承された値が上書きされます。

CASE_SNMP_READ_COMMUNITY

エージェントへ **GET** 要求を送信できる **SNMP** 読み取りコミュニティの名前を指定します。セミコロンで区切ることにより、複数のコミュニティを指定できます（たとえば **public1; public2**）。また、コミュニティごとにスペースで区切った **IP** アドレスリストを含めて、アクセスを制限できます（たとえば **public 1.2.3.4**）。この値も **CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。

CASE_SNMP_READ_COMMUNITY パラメータを使用して読み取りコミュニティを指定した場合は、継承された値が上書きされます。

デフォルト：**snmp_public**（新規インストールで（アップグレードは除く）、読み取り/書き込みコミュニティが指定されていない場合のみ有効）

CASE_SNMP_READ_ALLOWED_MANAGERS

CASE_SNMP_READ_COMMUNITY でエージェントへの照会を許可された **SNMP** マネージャの IP アドレス/ホスト名のスペース区切りリストを指定します。これを指定する場合は、**CASE_SNMP_READ_COMMUNITY** に 1 つの語 (**SNMP** コミュニティ) のみを含める必要があります。

CASE_SNMP_WRITE_COMMUNITY

エージェントへ **GET** 要求と **SET** 要求を送信できる **SNMP** 書き込みコミュニティの名前を指定します。セミコロンで区切るにより、複数のコミュニティを指定できます (たとえば **rwcomm1;rwcomm2**)。また、コミュニティごとにスペースで区切った IP アドレス リストを含めて、アクセスを制限できます (たとえば **rwcomm1 1.2.3.4**)。この値も **CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承できます。**CASE_SNMP_WRITE_COMMUNITY** パラメータを使用して書き込みコミュニティを指定した場合は、継承された値が上書きされます。

CASE_SNMP_WRITE_ALLOWED_MANAGERS

CASE_SNMP_WRITE_COMMUNITY でエージェントへの照会を許可された **SNMP** マネージャの IP アドレス/ホスト名のスペース区切りリストを指定します。これを指定する場合は、**CASE_SNMP_WRITE_COMMUNITY** に 1 つの語 (**SNMP** コミュニティ) のみを含める必要があります。

CASE_SNMP_TRAP_COMMUNITY

SNMP トラップ コミュニティおよびトラップ先アドレスを指定します。セミコロン (;) で区切ることにより複数のトラップ コミュニティ設定を指定できます。このパラメータの値は、**CASE_DEFAULT_FROM_NATIVE_SNMP** パラメータを使用して、ネイティブ **SNMP** エージェントから継承することもできます。**CASE_SNMP_TRAP_COMMUNITY** パラメータを使用してトラップ コミュニティを指定すると、継承された値が上書きされます。以下の値がこのパラメータに必要です。

- コミュニティ名
- トラップの送信先のデスティネーションアドレス

以下の値は、このパラメータのオプションです。

- トラップの送信先のポート番号
- トラップソースのエンコーディング オプション
- トラップソースのホスト名

トラップ コミュニティ設定の構文：

```
community-string {ip-address|hostname} [port [encoding [source]]]
```

例：

```
public 1.2.3.4;public 2.3.4.5 1162;trapcom 3.4.5.6 1162 100 4.5.6.7
```

CASE_SNMP_TRAP_DESTINATION

トラップの送信先のホスト名または IP アドレスを指定します。これを指定する場合は、**CASE_SNMP_TRAP_COMMUNITY** に 1 つの語 (**SNMP** コミュニティ) のみを含めると共に、**CASE_SNMP_TRAP_PORT** も指定する必要があります。

CASE_SNMP_TRAP_PORT

トラップの送信先のデスティネーション ポート番号を指定します。これを指定する場合は、**CASE_SNMP_TRAP_COMMUNITY** に 1 つの語 (**SNMP** コミュニティ) のみを含めると共に、**CASE_SNMP_TRAP_DESTINATION** も指定する必要があります。

CASE_DISABLE_NATIVE_SNMP

ネイティブ **SNMP** エージェントを停止して無効にするかどうかを指定します。

デフォルト： **no**

CASE_DEFAULT_FROM_NATIVE_SNMP

ネイティブ **SNMP** エージェントのデフォルトの **SNMP** 設定を使用するかどうかを指定します。

デフォルト : **no**

CASE_MANAGER_HOSTNAME

このエージェントを管理する構成マネージャのホスト名を指定します。このパラメータに値を入力すると、指定したマネージャからこのエージェントを設定できます。アスタリスク (*) を入力すると、エージェント システムを最初に検出したマネージャが受け入れられます。このマネージャは、エージェントの設定をフル コントロールできるようになります。デフォルトでは、マネージャ ホストは入力（使用）されず、エージェントは管理対象外モードで実行されます。

CASE_MANAGER_POLICY_NAME

エージェントで使用する必要がある、構成マネージャのポリシー ファイルの名前を指定します。このパラメータに値を入力すると、マネージャの既存の設定ファイルに従って **SystemEDGE** が設定されます。デフォルトでは、エージェントはインストール済みのポリシー ファイルを使用します。

CASE_START_AFTER_INSTALL

インストールが完了した後に、エージェントを自動的に開始するかどうかを指定します。有効な値は **Yes**、**No**、**PRESERVE** です。**PRESERVE** を使用すると、アップグレード時に、インストールを開始した時点でエージェントが実行されていた場合にのみエージェントを起動できます。

デフォルト : **PRESERVE**

CASE_PRIVSEP_USER

SNMP 通信中にエージェントが使用して実行する認証情報のユーザ名を指定します。

このエントリは、別のユーザ アカウントで **SNMP** 通信を実行するようにエージェントに指示します (**UNIX** のみ)。このエージェントはまた、有効なグループとして、このユーザのデフォルト グループを使用します。

デフォルト : エージェントは **root** アカウントを使用して動作します。

CASE_INSTALL_DOCS

SystemEDGE のドキュメントをエージェントと同時にインストールするかどうかを指定します。

デフォルト : **yes**

CASE_LEGACY_MODE

エージェントをレガシー モードでインストールするかどうかを指定します。レガシー モードでは、ベース エージェントのみがインストールされ、**CA Virtual Assurance** でのエージェントの使用を円滑にする要素は、まったくインストールされません。 **CA Virtual Assurance** でエージェントを使用しない場合は、レガシー モードを使用してインストールしてください。

エージェントを管理対象モードに変更するには、エージェントを再インストールまたはアップグレードし、**CASE_LEGACY_MODE=no** と指定します。

デフォルト : **no**

インストールが始まります。使用許諾契約に同意しなかった場合、インストールは失敗します。

lsm のインストーラ ログ ファイルは常に

`/opt/CA/installer/log/$CA_SETUP_PACKAGE_NAME.log` にあります。このファイルを使用して、インストールが成功したかどうかを確認できます。

インストールを確認する場合は、インストール ディレクトリに **SystemEDGE** のファイルがあるかどうかを確認します。

\$CASYSEDGE 変数のレガシー サポート

Linux/UNIX 上のインストール プログラムは、`/etc/profile` 内のシステム環境設定を変更しません。その結果、**\$CASYSEDGE** 変数は使用できなくなりました。

使用環境で **\$CASYSEDGE** をサポートする必要がある場合は、以下のいずれかのオプションを実行します。

- シェルで **\$CASYSEDGE** を作成する。

sh、ksh、または bash シェルで以下のコマンドを実行します。

```
./etc/profile.CA
```

csh を使用する場合は、以下のコマンドを実行します。

```
source /etc/csh_login.CA
```

- インストールプログラムが `/etc/profile` を変更し、`$CASYSEDGE` を作成するようにします。

シェルを開き、以下のコマンドを入力します。

```
Update_Profile=1;export Update_Profile  
sh ca-setup.sh
```

`csch` を使用する場合は、以下のコマンドを実行します。

```
setenv Update_Profile 1  
sh ca-setup.sh
```

インストールプログラムは **SystemEDGE** をインストールし、環境内に `$CASYSEDGE` を作成します。

リリース 5.7.1 へのアップグレード中に、**SystemEDGE** インストールプログラムは既存の環境を変更しません。以前に作成された `$CASYSEDGE` 変数は環境内に残ります。

応答ファイルの設定と使用

ユーザ操作のないサイレント インストールを実行するための応答ファイルを作成できます。 応答ファイルの使用は、コマンドラインに **CA_SETUP_MODE=UNATTENDED** を指定するのと同じ効果があります。 応答ファイルを使用すると、インストールはサイレント（無人）モードになります。

インストーラは、応答ファイル内のプロパティを使用して、ユーザ入力を要求せずにエージェントをインストールします。

次の手順に従ってください：

1. 管理者または **root** としてコンピュータ システムにログオンします。
2. 「コマンドラインを使用した～へのエージェントのインストール」セクションで指定したパラメータに基づいて応答ファイルを作成します。 応答ファイルは、以下のようなパラメータ設定で構成されるテキストファイルです。

```
parameter1=value1
parameter2=value2
...
```

3. **DVDdrive¥Installers¥OperatingSystem¥Agent¥SysMan¥CA_SystemEDGE_Core directory** ディレクトリに移動し、使用中のオペレーティング システムに従って以下のいずれかのコマンドを入力します。

```
ca-setup CA_SETUP_RESPONSE_FILE="<name of the response file>" (Windows)
```

```
sh ca-setup.sh CA_SETUP_RESPONSE_FILE="<name of the response file>" (UNIX, Linux)
```

CA_SETUP_RESPONSE_FILE

応答ファイルのパスと名前を指定します。

応答ファイルの設定を使用して、サイレント モードでインストールが実行されます。

レガシー モードでのエージェントのインストール

レガシー モードで SystemEDGE をインストールすると、ベース エージェントのすべての機能が得られますが、CA Virtual Assurance と組み合わせて使用するときに役立つ以下のコンポーネントは除外されます。

- CA Virtual Assurance からのリモート設定を可能にする CAM
- CA Virtual Assurance からのリモート展開を可能にする IDPrimer

CA Virtual Assurance または同様の管理アプリケーションで SystemEDGE を管理することを予定してしない場合にのみ、SystemEDGE をレガシー モードでインストールしてください。レガシー モードでエージェントをインストールした後、CA Virtual Assurance を使用してエージェントを管理したい場合は、エージェントをアップグレードできます。

注: SystemEDGE の旧バージョンをアップグレードするときは、特に指定しない限り、完全なエージェントに自動的にアップグレードされます。

次の手順に従ってください:

1. CA_SystemEDGE_Core ディレクトリをインストール メディアからハードディスクにコピーします。
2. CA_SystemEDGE_Core ディレクトリに移動し、ASCII エディタで ca-setup.dat を開きます。
3. ca-setup.dat を編集して、CASE_LEGACY_MODE=yes を設定します。
4. ca-setup.dat を保存します。
5. 「[エージェントの Windows へのインストール \(P. 94\)](#)」または「[エージェントの UNIX および Linux システムへのインストール \(P. 108\)](#)」の説明に従ってインストールを実行します。
6. インストールを完了します。

コマンドラインからレガシー モードでエージェントをインストールするには、ca-setup コマンドに以下のパラメータを追加します。

```
CASE_LEGACY_MODE="yes"
```

エージェントの展開

CA Virtual Assurance は、SystemEDGE エージェントをすべての管理対象システムにリモート展開するための包括的なソリューションを提供します。カスタマイズされたインストールパラメータを含む付属パッケージに基づいて展開テンプレートを作成すると同時に、そのようなテンプレートを多数の管理対象システムに展開できます。この自動展開ソリューションによって、1つの場所から、企業全体にわたってエージェントを展開して設定することが可能となります。

CA Virtual Assurance には、以下のベース展開パッケージが用意されています。

- SystemEDGE Agent Core
- SystemEDGE Advanced Encryption
- CA Citrix XenServer AIM
- SystemEDGE リモート モニタリング AIM
- SystemEDGE サービス レスポンス モニタ AIM
- CA Systems Performance LiteAgent
- CA IBM LPAR AIM
- CA IBM High Availability Cluster Multiprocessing AIM
- CA KVM AIM (CA Server Automation には適用できません)
- CA Cisco UCS AIM
- CA Microsoft Hyper-V AIM
- CA Microsoft Cluster Service Support AIM
- CA Solaris Zones AIM
- CA VMware vCenter Server AIM
- CA VMware vCloud AIM
- Exchange Server および Active Directory AIM (CA Server Automation には適用できません)

CA Virtual Assurance は、以下の SystemEDGE 展開シナリオをサポートします。

- SystemEDGE エージェントがまだ存在しないシステムに SystemEDGE リリース 5.7.1 エージェントを展開できます。
- SystemEDGE 4.3 エージェントが存在するシステムに SystemEDGE リリース 5.7.1 エージェントを展開できます。展開すると、既存のエージェントがリリース 5.7.1 に自動的にアップグレードされます。
- 個々のシステムに対する設定変更または設定テンプレートによる設定変更を行うことができます。後者のオプションでは、変更を既存の管理対象 SystemEDGE エージェントまたは管理対象 SystemEDGE エージェントのグループに展開できます。

注: 展開機能の詳細については、*CA Virtual Assurance* のオンラインヘルプおよび「管理ガイド」を参照してください。

CA Virtual Assurance は、以下の SystemEDGE 展開シナリオをサポートしません。

- SystemEDGE 5.0.0 より古いバージョンのエージェントは展開できません。
- SystemEDGE エージェントを CA Virtual Assurance マネージャ システム上に展開することはできません。エージェントは CA Virtual Assurance マネージャによって自動的にインストールされます。

注: 展開のサポートの詳細については、「CA Virtual Assurance 管理ガイド」を参照してください。

CA Virtual Assurance によるエージェントの展開

「実装ガイド」およびオンラインヘルプでは、ユーザインターフェースからエージェントのパッケージを展開する方法の詳細について説明しています。

システムにエージェントを展開するには、展開ジョブを作成します。展開ジョブには、展開パッケージを適切なシステムに適切なタイミングで配信するために、CA Virtual Assurance にとって必要な詳細が含まれます。

次の手順に従ってください:

1. [リソース] - [展開] を選択します。
[展開] ペインに [パッケージ]、[テンプレート]、および [ジョブ] が表示されます。
2. [管理対象リソース] ペインで [ジョブ] フォルダを右クリックし、[新規ジョブの作成] を選択します。[ジョブ] フォルダを選択し、[ジョブ ステータス] ツールバーの [+] (新規) をクリックする方法もあります。
[ジョブ セットアップ] ページが表示されます。

3. [ジョブ名] ペインで名前を入力し、オプションで、ベースにする既存のテンプレートを選択して、[次へ] をクリックします。
[パッケージ選択] ページが表示されます。
4. プラットフォームと、展開するパッケージを選択します。
5. (オプション) [詳細] タブをクリックします。
[パッケージラッパー詳細] ダイアログ ボックスが表示され、パッケージプロパティをインラインで編集できます。パッケージラッパーが不完全または無効な状態であっても、フィールドがインライン編集によって修正できる場合。
 - a. [編集] をクリックし、パッケージラッパーのプロパティを変更します。
 - b. [保存] をクリックし、[OK] をクリックします。
パッケージラッパーのプロパティが更新されます。
6. 下矢印をクリックしてパッケージラッパーをジョブに追加し、[次へ] をクリックします。
[マシン選択] ページが表示されます。
7. 展開先のシステムを選択し、[次へ] をクリックします。環境内に多数のサーバがある場合、すべてのサーバを一覧表示するには、一定数のエントリを含むページが複数必要になることがあります。 ページでサーバを選択し、次のページにスクロールしても、前のページで行った選択内容は有効なままです。
[選択済みマシン] ページが表示されます。
8. [認証情報の設定] をクリックし、接続を確立するために必要なシステム認証情報を設定して、[次へ] をクリックします。
注: ドメイン認証情報を使って **Windows** ターゲット システムに展開する場合は、「**DOMAIN¥ユーザ名**」の形式を使用する必要があります。
詳細設定ページが表示されます。
9. (オプション) 展開を管理する配布サーバを設定します。設定しない場合は、自動的に選択されます。

10. ジョブのスケジュール オプションを選択します。

即時配布

新しい展開ジョブを作成した直後にジョブを開始します。即時配布はデフォルト オプションです。

時差配布

特定の時間にパッケージを配布します。

スケジュール済み配布

将来の特定の時間に展開をスケジュールします。

11. (オプション) 以前に同じ展開インフラストラクチャを使ってパッケージをシステムに正常に展開したことがある場合は、再度そのインフラストラクチャを強制的に実行することができます。
12. [次へ] をクリックします。
[サマリ] ページが表示されます。
13. ジョブの詳細を確認し、[展開] をクリックします。
展開ジョブが作成されます。

注: 作成したジョブはテンプレートとして保存できます。将来のジョブで簡単に再利用できるように、テンプレートにはパッケージとマシンの選択内容が保存されます。詳細については、オンラインヘルプおよび「実装ガイド」を参照してください。

SystemEDGE のアンインストール

このセクションでは、SystemEDGE エージェントに関連付けられたファイルとサブディレクトリを削除する方法について説明します。

Windows 上の SystemEDGE および AIM のアンインストール

アンインストーラは、システムから SystemEDGE を削除します。ユーザは、データディレクトリからコンフィギュレーションデータを削除するかどうかを指定できます。Windows の [プログラムの追加と削除] ウィンドウまたはコマンドラインを使用してエージェントと AIM をアンインストールできます。

アンインストールするときは、以下の依存関係を考慮してください。

- 仮想環境用の AIM、RM AIM、および SRM AIM は Advanced Encryption と SystemEDGE に依存します。
- Advanced Encryption は SystemEDGE に依存します。

これらの依存関係に基づいて、アンインストールの順序は以下のようになります。

1. RM AIM、SRM AIM、または仮想環境用の AIM
2. Advanced Encryption
3. SystemEDGE コア

これ以外の順序ではアンインストールできません。リモート展開でインストールされたコンポーネントを削除する場合は、Idprimer と CAM はアンインストールされません。

SystemEDGE または AIM を Windows の [プログラムの追加と削除] ウィンドウでアンインストールする方法

1. [スタート] - [設定] - [コントロール パネル] - [プログラムの追加と削除] を選択します。

[プログラムの追加と削除] ウィンドウが表示され、以下のコンポーネントが一覧表示されます。

- 仮想環境用の AIM
- CA SystemEDGE Core
- CA SystemEDGE AdvancedEncryption
- CA SystemEDGE RM
- CA SystemEDGE SRM

2. アンインストールの順序に従って適切なコンポーネントを右クリックし、[アンインストール] を選択します。

SystemEDGE の場合は、設定ファイルを保持するかどうかを指定するダイアログボックスが表示されます。

3. [はい] または [いいえ] をクリックします。

アンインストールプロセスのチャートを示すダイアログ ボックスが表示されます。アンインストールが完了すると、ダイアログ ボックスが閉じます。

コマンドラインを使用して SystemEDGE または AIM をアンインストールする方法

1. コマンドプロンプトを開き、DVD1¥Installers¥Windows¥Agent¥SysMan ディレクトリに移動します。以下のサブディレクトリが含まれています。

- CA_SystemEDGE_SRM
- CA_SystemEDGE_RM
- CA_SystemEDGE_AdvancedEncryption
- CA_SystemEDGE_Core

2. 以下のコマンドを実行します。

```
ca-setup -x
```

SystemEDGE の場合は、設定ファイルを保持するかどうかを指定するダイアログ ボックスが表示されます。

3. [はい] または [いいえ] をクリックします。

アンインストールプロセスのチャートを示すダイアログ ボックスが表示されます。アンインストールが完了すると、ダイアログ ボックスが閉じます。

SystemEDGE または AIM のサイレント アンインストールを実行する方法

- SystemEDGE をアンインストールするには、以下の手順に従います。
 - a. コマンドプロンプト ウィンドウを開き、以下のディレクトリ パスに移動します。

```
DVD1¥Installers¥Windows¥Agent¥SysMan¥CA_SystemEDGE_Core
```

- b. 以下のコマンドを実行します。

```
ca-setup.exe CA_SETUP_MODE=UNINSTALL CASE_KEEP_DATA=[YES|NO]
```

SystemEDGE がコンピュータからアンインストールされます。

注: 確認するには、コントロール パネルに移動し、SystemEDGE がコントロール パネルの項目から削除されていることを確認します。

- AIM を削除するには、以下の手順に従います。
 - a. コマンドプロンプト ウィンドウを開き、以下の適切な AIM ディレクトリ パスに移動します。

```
DVD1¥Installers¥Windows¥Agent¥SysMan¥AIM
```

- b. 以下のコマンドを実行します。

```
ca-setup.exe CA_SETUP_MODE=UNINSTALL
```

AIM がコンピュータからアンインストールされます。

注: [Program Files] - [CA] - [SystemEDGE] - [plugins] に移動し、AIM フォルダが削除されていることを確認してください。

UNIX システム上の SystemEDGE および AIM のアンインストール

アンインストーラは、システムから SystemEDGE または AIM を削除します。SystemEDGE については、データ ディレクトリから設定データを削除するかどうかを指定できます。

アンインストールするときは、以下の依存関係を考慮してください。

- SRM AIM は Advanced Encryption および SystemEDGE に依存します。
- Advanced Encryption は SystemEDGE に依存します。

これらの依存関係に基づいて、アンインストールの順序は以下のようになります。

1. SRM AIM
2. Advanced Encryption
3. SystemEDGE コア

これ以外の順序ではアンインストールできません。リモート展開でインストールされたコンポーネントを削除する場合は、Idprimer と CAM はアンインストールされません。

ca-setup.sh を使用してエージェントまたは AIM をアンインストールする方法

1. 端末コンソールを開き、root としてログイン (su) します。
2. DVD2/Installers/<Platform>/Agent/SysMan ディレクトリに移動します。以下のサブディレクトリが含まれています。
 - CA_SystemEDGE_SRM
 - CA_SystemEDGE_AdvancedEncryption
 - CA_SystemEDGE_Core
3. 適切なディレクトリに移動し、以下のコマンドを実行します。

```
sh ca-setup.sh -x
```

設定ファイルを保持するかどうかを指定するダイアログ ボックスが表示されます。

4. [はい] または [いいえ] をクリックします。

アンインストール プロセスのチャートを示すダイアログ ボックスが表示されます。アンインストールが完了すると、ダイアログ ボックスが閉じます。

lsm を使用してエージェントまたは AIM をアンインストールする方法

1. 端末コンソールを開き、**root** としてログイン (**su**) します。
2. 以下のリストから適切なコマンドを選んで実行します。

```
lsm -e CA_SystemEDGE_SRM  
lsm -e CA_SystemEDGE_AdvancedEncryption  
lsm -e CA_SystemEDGE_Core
```

SystemEDGE の場合は、設定ファイルを保持するかどうかを指定するように求められます。

3. [はい] または [いいえ] をクリックします。

アンインストールが完了します。

エージェントまたは AIM のサイレント アンインストールを実行する方法

- エージェントをアンインストールするには、以下の手順に従います。
 - a. 端末コンソールを開き、**root** としてログイン (**su**) します。以下のディレクトリ パスに移動します。
DVD2/Installers/Platform/Agent/SysMan/CA_SystemEDGE_Core
 - b. 以下のコマンドを実行します。

```
sh ca-setup.sh CA_SETUP_MODE=UNATTENDED_UNINSTALL CASE_KEEP_DATA=[YES|NO]
```

エージェントがコンピュータからアンインストールされます。
- AIM をアンインストールするには、以下の手順に従います。
 - a. 端末コンソールを開き、**root** としてログイン (**su**) します。以下の適切な AIM ディレクトリ パスに移動します。
DVD2/Installers/Platform/Agent/SysMan/AIM
 - b. 以下のコマンドを実行します。

```
sh ca-setup.sh CA_SETUP_MODE=UNATTENDED_UNINSTALL
```

AIM がコンピュータからアンインストールされます。

再インストール

SystemEDGE を再インストールするときは、インストーラは以下のパラメータのみを使用します。

CASE_START_AFTER_INSTALL

CASE_SNMP_PORT

CASE_INSTALL_DOCS

その他のパラメータは無視されます。

管理対象ノードおよび AIM サーバのアップグレード

管理対象ノードおよび AIM サーバをアップグレードする推奨の方法は、リモート展開を使用することです。リモート展開を使用すると、SystemEDGE、Advanced Encryption、および AIM をアップグレードできます。

ご使用のシステムで SystemEDGE の管理対象モードがサポートされていない場合、リモート展開は使用できません。以下のいずれかのオプションを使用することをお勧めします。

- インストールメディアからの手動インストール
- たとえば SSH などを使用した、SystemEDGE、Advanced Encryption、および SRM AIM のリモートインストール

以下の手順に従います。

[エージェントと AIM のアップグレード](#) (P. 136)

[ポリシーへの SystemEDGE モニタのインポート](#) (P. 139)

エージェントと AIM のアップグレード

SystemEDGE リリース 5.7.1 へは、SystemEDGE の以下のアップグレード可能リリースからアップグレードできます。

- 4.3.4 以降 (4.3.x)
- 5.1.0 以降 (5.1.x)
- 5.6.0 以降 (5.6.x)
- 5.7.0 以降 (5.7.x)

インストーラは元のインストールパスのルートにアップグレードされたディレクトリを作成し、前のリリースからそのディレクトリに設定ファイルをコピーします。エージェントは、初めて起動したときに設定データを新しく作成されたデータ ディレクトリにマイグレートします。SystemEDGE をリリース 5.7.1 にアップグレードする場合は、Advanced Encryption および対応する AIM をすべて最新バージョンにアップグレードしてください。

注: SystemEDGE リリース 5.7.1 は、前の CA Virtual Assurance リリースの AIM をロードしません。

以下のいずれかの方法で SystemEDGE をアップグレードできます。

- アップグレード可能なエージェントがインストールされたシステムに対する CA Virtual Assurance によるリモート展開

注: SystemEDGE、Advanced Encryption、および AIM をアップグレードする場合は、アップグレード用の単一のリモート展開ジョブを使用します。

SystemEDGE、Advanced Encryption、およびすべての必要な AIM を [リモート展開] タブに追加します。SystemEDGE は、5.7.1 レベルの AIM または iddmod のようなレガシー AIM のみをロードします。したがって、5.7 レベル以下の AIM は隔離されます。

リモート展開の詳細については、「[管理ガイド](#)」の「リモート展開」の章を参照してください。

- Windows でのカスタム マネージャ インストール。このオプションはインストール済みコンポーネントをすべてアップグレードします。インストールする他のエージェントおよび AIM をすべて選択します。
- アップグレード可能なエージェントがインストールされたシステムに対する手動インストール

アップグレードを実行するには、「[Windows へのエージェントのインストール \(P. 94\)](#)」または「[UNIX へのエージェントのインストール \(P. 108\)](#)」の説明に従ってエージェントをインストールします。

アップグレードがサポートされていない古いリリースの場合は、以下の手順に従います。

- 以前のリリースの SystemEDGE の場合は、エージェントをアンインストールしてから SystemEDGE リリース 5.7.1 をインストールします。アンインストールの前にすべての設定データを保存し、そのデータを SystemEDGE リリース 5.7.1 に適用できます。

関連項目：

[SystemEDGE 4.3.4 からのエージェントアップグレード \(P. 137\)](#)

SystemEDGE 4.3.4 からのエージェントアップグレード

インストーラは、設定ファイルを以前の 4.3.4 リリースからアップグレードするとき以下を実行します。

- インストーラは、以下のファイルの旧バージョンを検出して、CASYSEDGE¥upgraded ディレクトリにコピーします。
 - system32¥sysedge.cf (Windows) および /etc/sysedge.cf (UNIX)
 - CASYSEDGE¥config¥sysedgeV3.cf (Windows および UNIX)
 - system32¥sysedge.mon (Windows) および /etc/sysedge.mon (UNIX)
 - system32¥sysedge.lic (Windows) および /etc/sysedge.lic (UNIX)
 - CASYSEDGE¥plugins¥monwin¥monwin.cf (Windows および UNIX)
- エージェントを初めて起動すると、以下のファイルが CASYSEDGE_DATADIR ディレクトリに直接コピーされます。
 - CASYSEDGE¥config¥sysedge.cf (Windows および UNIX)
 - CASYSEDGE¥config¥sysedgeV3.cf (Windows および UNIX)
- エージェントは、sysedge.cf のコピー操作の一部として、アップグレードされるディレクトリに含まれる sysedge.cf、sysedge.mon、および monwin.cf ファイル内の設定データをデータ ディレクトリ内の新しい sysedge.cf ファイルにマイグレートします。
- エージェントは、sysedgeV3.cf のコピー操作の一部として、アップグレードされるディレクトリに含まれる sysedgeV3.cf ファイル内の設定データをデータ ディレクトリ内の新しい sysedgeV3.cf ファイルにマイグレートします。

注：データ ディレクトリには、実行時の設定変更を使用される sysedge.cf ファイルのバージョンを格納します。

エージェントは、以前のリリースからアップグレードするときに、設定ファイルの設定に対して自動的に以下の変更を加えます。

- **procAlive** プロセス モニタの構文は、**threshold** プロセス モニタの構文に合わせて変更されます。エージェントは、既存の **procAlive** エントリを自動的に新しい形式に変換します。

注: **procAlive** エントリの構文の詳細については、「プロセスとサービスのモニタリング」の章を参照してください。

- **no_process_sets** および **no_remoteshell_group** パラメータを無効にすると、SNMP 書き込みコミュニティのみを使用してエージェントシステムへのクリティカルなアクセスが可能になります。このため、エージェントはこれらのパラメータの古い設定をマイグレートせず、これらのパラメータを常に有効にします。
- **sysedge_memory** パラメータは廃止され、アップグレードされた **sysedge.cf** ファイルから削除されます。未処理のアラームは、**sysedge.mon** ファイルにモニタの現在の状態を格納することによって処理されるようになりました。既存のエントリは、新しいアラーム処理設定にマイグレートされます。
- **tc_publish** パラメータの名前は（真偽が逆の）**no_trapcommunity_table** に変更され、デフォルトで有効になります。
- リモート展開を使用して **SystemEDGE** リリース 4.3.4 から **SystemEDGE** リリース 5.7.1 にアップグレードすると、マイグレートされたモニタは **sysedge.cf.bak** ファイルに格納されます。
- 展開の戦略によっては、[ポリシー設定] を使用してマイグレートされたモニタを環境に適用できます。

注: 詳細については、**CA Virtual Assurance** のマニュアル選択メニューを参照してください。

- ローカル システム上の **SystemEDGE** リリース 4.3.4 を **SystemEDGE** リリース 5.7.1 にアップグレードすると、インストールプログラムはモニタをマイグレートし、それを新しい設定に反映します。

以前の構文はすべて新バージョンのエージェントと互換性がありますが、エージェントの機能を強化する多くのオプションが新たに追加されています。古いモニタエントリを調べ、構文を編集して追加のオプションを組み込むことをお勧めします。

注: SystemEDGE をリリース 4.3.4 以降 (4.3.x) からアップグレードする場合、インストーラは以下のパラメータのみを使用します。

```
CASE_PUBDATADIR
CASE_MANAGER_HOSTNAME
CASE_MANAGER_POLICY_NAME
CASE_START_AFTER_INSTALL
CASE_LEGACY_MODE
CASE_SNMP_PORT
CASE_INSTALL_DOCS
CASE_SNMP_TRAP_COMMUNITY (1)
CASE_SNMP_TRAP_DESTINATION (1)
CASE_SNMP_TRAP_PORT (1)
CASE_SNMP_READ_COMMUNITY (1)
CASE_SNMP_WRITE_COMMUNITY (1)
CASE_SNMP_READ_ALLOWED_MANAGERS (1)
CASE_SNMP_WRITE_ALLOWED_MANAGERS (1)
```

その他のパラメータは無視されます。

(1) これらのパラメータは特別です。これらの設定は、既存の SystemEDGE 4.x 設定に追加され、これによって、SystemEDGE 4.x マネージャと SystemEDGE 5.x マネージャの両方が機能するようになります。

ポリシーへの SystemEDGE モニタのインポート

エージェントをアップグレードする前に、SystemEDGE エージェントで使用される SNMP コミュニティ文字列が [管理] タブの [SNMP] で指定されていることを確認します。コミュニティ文字列はコンピュータごとに指定することもできます ([ポリシー] - コンピュータ - [メトリック] - [SNMP])。SystemEDGE インストーラは SystemEDGE ファイルを「アップグレード後の」ディレクトリに移動します。アップグレード後、エージェントはこれらのファイルを読み取ります。OID は保持されます。ポリシー設定では既存の sysedge.cf ファイルのエントリが読み取られるだけです。sysedge.mon ファイルはインポートされません。

リモート展開では、モニタの **Raw OID** がインスタンス内にインポートされます。

次の手順に従ってください:

1. **SystemEDGE** エージェントのコミュニティ文字列が、[管理] タブで、エージェントのポートの **SNMP** に含まれていることを確認します。
2. **SystemEDGE** をリリース **5.7.1** にアップグレードします。
3. **SystemEDGE** リリース **5.7.1** が正しく検出されることと（[リソース] タブ）、このリリースにポリシーがあることを確認します。
4. [リソース] タブをクリックして [設定] ペインを開き、[ポリシー] を展開して [SystemEDGE] をクリックします。
[SystemEDGE] ペインが表示されます。
5. [利用可能ポリシー] ツールバーの [+]（新規）をクリックします。
[新規 SystemEDGE ポリシー] ダイアログ ボックスが表示されます。
6. ポリシーの名前を入力し、[インポート] をクリックします。
[SystemEDGE エージェント マシン] ダイアログ ボックスが表示されます。
7. アップグレードされたサーバを選択し、[OK] をクリックします。
[SystemEDGE エージェント マシン] ダイアログ ボックスが閉じます。
8. [新規 SystemEDGE] ダイアログ ボックスの [OK] をクリックします。
インポートしたモニタを含むポリシーが **CA Virtual Assurance** によって作成されます。

モニタは編集または更新できます。**SystemEDGE** リリース **5.7.1** 状態モデルを使用することもできます。

第 5 章: エージェントの設定

この章では、SystemEDGE エージェントの動作を決定する設定パラメータの設定方法について説明します。これらの設定パラメータは、設定ファイル `sysedge.cf` および `sysedgeV3.cf` に定義されています。

このセクションには、以下のトピックが含まれています。

[設定の基本](#) (P. 141)

[sysedge.cf と sysedge.mon のやり取り](#) (P. 143)

[CA Virtual Assurance を使用した設定](#) (P. 143)

[ファイルベースの設定](#) (P. 144)

[モニタおよび自動ウォッチャーの正規表現の例](#) (P. 175)

[Perl 互換正規表現 \(PCRE\) のサポート](#) (P. 178)

[SNMPv3 設定](#) (P. 185)

[Windows の SystemEDGE コントロールパネル](#) (P. 196)

設定の基本

SystemEDGE のすべての設定情報は、`sysedge.cf` および `sysedgeV3.cf` 設定ファイルに格納されています。エージェントを設定するには、CA Virtual Assurance のユーザインターフェース内の設定を使用するか、または直接操作によって `sysedge.cf` ファイルを変更します。

`sysedge.cf` および `sysedgeV3.cf` 設定ファイルには、SystemEDGE のすべてのエージェント設定情報が格納されています。

`sysedge.cf` 設定ファイルは、テキストエディタで表示可能な UTF-8 でエンコードされた (プレーン) テキストファイルです。このファイルには、以下のようなローカルシステムの値が格納されています。

- システムの説明、コミュニティ文字列、およびトラップコミュニティ
- 機密情報のレポートやアクションスクリプトの実行に関するエージェントの動作
- エージェントのモニタリングテーブルのエントリ
- AIM のロードパラメータと設定パラメータ

`sysedgeV3.cf` 設定ファイルは非暗号化ファイルですが、`se_enc` ユーティリティを使用して暗号化できます。このファイルには、以下のような **SNMPv3** の情報が格納されています。

- **SNMPv3 ユーザの設定**
- **SNMPv2c および SNMPv3 のトラップ デスティネーション**

注:

- `sysedge.cf` または `sysedgeV3.cf` ファイルを手動で変更したときは、変更を有効にするために **SystemEDGE** エージェントを再起動する必要があります。
- `sysedge.mon` ファイルは実行時に作成され、**SystemEDGE** エージェントによって自動的に更新されます。

UNIX および Windows プラットフォームでは、`sysedge.cf` および `sysedgeV3.cf` ファイルは `CASYSEDGE¥config` ディレクトリにインストールされます。`CASYSEDGE` は **SystemEDGE** が UNIX にインストールされるディレクトリのプレースホルダです。Windows では、この場所はレジストリ内の `HKLM¥Software¥ComputerAssociates¥SystemEDGE¥InstallDir` に格納されます。ユーザは、インストール時に **SNMPv1** コミュニティ、`sysContact`、エージェントの **SNMP** ポート、データディレクトリなどの情報を指定するように求められます。ユーザが指定した情報は、インストーラによって `sysedge.cf` ファイルに反映されます。

SystemEDGE エージェントは、初めて起動したときに `*.cf` ファイルをデータディレクトリにコピーします。このデータディレクトリ内のファイルを使用してエージェントを設定します。Windows システムでは、**SystemEDGE** コントロールパネルを介してこれらのファイルにアクセスすることもできます。これらのファイルにアクセスするには、[スタート]-[設定]-[コントロールパネル]-[**SystemEDGE**]を選択し、`sysedge.cf` または `sysedgeV3.cf` をクリックします。

インストーラは、インストール時にデフォルトのエージェントインスタンス (**SystemEDGE** サービス) 用のポートを設定します。コマンドラインからエージェントを起動するときは、デフォルトインスタンスに対して `-p` パラメータを指定しないでください。その場合は、自動的にデフォルトのポートが使用されます。デフォルトのエージェントインスタンスのみが **CA Virtual Assurance** マネージャと通信します。その結果、**CA Virtual Assurance** マネージャから受信した設定ファイル (`sysedge.cf`、`sysedgeV3.cf`) は常に `SystemEDGE¥data¥portdefault-port` ディレクトリ (たとえば、`C¥Program Files¥CA¥SystemEDGE¥data¥port161`) に配置されます。

エージェントは、アップグレード時に以前のバージョンの設定を追加でマージします。設定ファイルの設定をアップグレードする方法の詳細については、「[アップグレードにおける注意事項 \(P. 137\)](#)」を参照してください。

sysedge.cf と sysedge.mon のやり取り

SystemEDGE エージェントは、sysedge.cf ファイルに自動的に入力されない SNMP SET による変更を格納するためのバックアップとして sysedge.mon ファイルを使用します。このファイルには、モニタの現在の状態などのランタイム情報も格納されます。sysedge.cf 内のエントリは、sysedge.mon 内のエントリより優先されます。

エージェントは、完全な再起動（コールドリスタート）を実行すると、最初にデータ内の sysedge.mon ファイルを読み取り、次に sysedge.cf ファイルを読み取り、既存のモニタ（現在の状態の情報および notInService 行のステータスを除く）を上書きし、モニタを追加します。CA Virtual Assurance マネージャを使用してエージェントを再設定すると、エージェントは配信された sysedge.cf ファイルのエントリのみを読み取ります。

sysedge.cf から手動でモニタリング エントリを削除し、そのエントリが sysedge.mon 内にも存在する場合は、エージェントが使用しないように sysedge.mon のエントリも削除する必要があります。

CA Virtual Assurance を使用した設定

リモートで CA Virtual Assurance のファイルベースの設定機能を使用して SystemEDGE エージェントを設定できます。CA Virtual Assurance のユーザインターフェースから sysedge.cf ファイル内の任意のパラメータ（システム設定、セキュリティ設定、モニタ、AIM など）を設定し、一元化された場所から 1 つ以上のエージェント インスタンスに設定を展開できます。

CA Virtual Assurance を使用して設定ファイルを展開すると、エージェントは実行中に任意の変更を適用でき、ウォーム スタートを実行するだけで済みます。SystemEDGE の設定方法としては、CA Virtual Assurance によるリモート設定をお勧めします。

CA Virtual Assurance のユーザ インターフェースからエージェントを設定する方法の詳細については、「CA Virtual Assurance 管理ガイド」およびオンライン ヘルプを参照してください。CA Virtual Assurance で実装できるエージェントのセキュリティ ロック オプションの詳細については、「概念」の章を参照してください。

ファイルベースの設定

以下の方法でエージェントを設定できます。

- CA Virtual Assurance ポリシーを介してリモートでファイルベースの設定を使用する
- CA eHealth AdvantEDGE View、CA Spectrum、CA NSM Agent View、または SystemEDGE のコマンドラインユーティリティを介してリモートで SNMP SET を使用する
- sysedge.cf ファイルをローカルで直接更新する

CA Virtual Assurance を使用してリモートでエージェントを設定すると、CA Virtual Assurance マネージャは設定された sysedge.cf ファイルを指定されたエージェントインスタンス（複数可）の CASYSEDGE_DATADIR ディレクトリに送信します。このディレクトリの場所には、CA Virtual Assurance から展開されたあらゆるランタイム設定が格納され、完備されます。エージェントは、このディレクトリにアップロードされた新しいファイルを検出すると、ウォームスタートを完了し、変更を適用します。sysedge.cf ファイルが CA Virtual Assurance マネージャによってエージェントに配信されると、SNMP Set によって行われた変更は上書きされます。

SNMP Set を使用してエージェントを設定すると、変更が適用され、sysedge.mon ファイルに保存されます。

注: SNMP 情報やセキュリティ情報など、一部のシステム設定は SNMP Set を使用して設定できません。

ローカルの sysedge.cf ファイルを直接更新してエージェントを設定したときは、変更を有効にするためにエージェントを再起動する必要があります。

以下のトピックでは、sysedge.cf ファイル内のパラメータと、各パラメータで利用可能な設定オプションについて説明します。CA Virtual Assurance を使用した設定手順については、「CA Virtual Assurance 管理ガイド」およびオンラインヘルプを参照してください。

sysedge.cf ファイル内のデフォルトエントリは、名前の先頭にテンプレートディレクティブが付いています。テンプレートディレクティブは、CA Virtual Assurance を使用してエージェントを設定したときの実際の設定エントリの作成を制御します。CA Virtual Assurance を使用してエージェントを設定すると、実際の設定エントリはファイルを展開したときに自動的に作成されます。

`sysedge.cf` ファイルを手動で編集するときは、テンプレートディレクティブが付いていない設定エントリを手動で入力し、パラメータ設定を実装する必要があります。各エントリはファイル内の関係する領域に記述され、グループ化されます。必要なエントリを適切な領域のテンプレートディレクティブの下に手動で入力します。

`sysedge.cf` ファイルを手動で編集するときは、データディレクトリ内の `sysedge.cf` ファイルを使用します。

システム情報の設定

インストール時にシステムの説明、場所、および担当者の情報を定義できます。これらの情報は、MIB-II 内のシステムグループに入力されます。これらの値は、インストール後も `sysedge.cf` ファイルで変更できます。

システム情報を設定するには、`sysedge.cf` ファイルで以下のフィールドを入力または（インストール時に設定を入力した場合は）更新します。

```
sysdescr <description>
syscontact <contact>
syslocation <location>
```

SNMP 情報の設定

`sysedge.cf` ファイルには、SystemEDGE エージェントによる SNMP 通信の方法を指示する SNMP アクセス情報および設定情報が含まれています。

`sysedge.cf` ファイルには以下の SNMP 情報を設定できます。

注: 一部の情報は、エージェントのインストール時に入力された情報に基づいて設定されます。変更が必要な場合は、ファイル内のこれらの情報を編集できます。

- SNMP 通信用の IP 方式
- SNMP バインドアドレス
- アクセス コミュニティ
- SNMPv1 トラップ コミュニティ
- SNMPv1 トラップ ソース
- SNMPv1 トラップ デスティネーション

SNMP User Datagram Protocol (UDP)通信用の IP ファミリの設定

ホストには、複数の IP ファミリ ソース ソケットと複数のファミリ デスティネーション アドレスを設定できます。また、エージェントの **SNMP User Datagram Protocol (UDP)** 通信の優先方式を設定できます。

UDP 通信の優先される IP ファミリ方式を設定するには、`sysedge.cf` ファイルに以下の行を追加します。

```
sysedge_ip_family <method>
```

method

優先される通信方式を指定します。指定した方式が利用可能でない場合、エージェントは **IPv4** を使用し、次に **IPv6** を使用します。以下のオプションを使用できます。

1

エージェントが優先される **SNMP UDP** 通信方式として **IPv4** のみを使用することを指定します。

2

エージェントが優先される **SNMP UDP** 通信方式として **IPv6** のみを使用することを指定します。

3

エージェントが優先される **SNMP UDP** 通信方式として **IPv4** と **IPv6** の両方を使用することを指定します。これはデフォルトの方式です。

例

SNMP UDP 通信で **IPv4** モードのみを（利用可能な場合に）使用するには、以下の行を入力します。

```
sysedge_ip_family 1
```

SNMP バインドアドレスの設定

デフォルトでは、SystemEDGE はすべてのインターフェース（*/UDP-161）にバインドします。以下のように `sysedge.cf` ファイルで `bind_address` パラメータを使用することにより、エージェントを特定のインターフェースにバインドできます。

```
bind_address <address>
```

address

エージェントをバインドする特定のインターフェースを指定します。
この値は、IPv4 アドレスまたは IPv6 アドレスです。

例

`sysedge.cf` ファイルに以下の行を入力すると、アドレス 10.1.0.202 のみにバインドします。

```
bind_address 10.1.0.202
```

`sysedge.cf` ファイルに以下の行を入力すると、アドレス ea2f:fe90:abcd:0000:230:a2f:200:ad01 のみにバインドします。

```
bind_address ea2f:fe90:abcd:0000:230:a2f:200:ad01
```

バインドアドレスの設定

SystemEDGE のバインドアドレスを設定する際は、以下のような場合を考慮します。

- SystemEDGE インストーラが、インストール時にバインドアドレスの設定をサポートしない。
- ポリシー設定では、個別のマシン レベルではなく、ポリシー レベルでバインドアドレスを定義する必要がある。

ポリシー設定では、バインドアドレスをポリシー内で設定する必要があります。バインドアドレスが管理または設定をプッシュしているエージェントより前に設定されている場合、バインドアドレスは元の設定から取得され、配信された設定に自動的に含まれます。このプロセスを実行できる方法はいくつかあります。

以下の用語がこれらのユース ケース全体で使用されます。

静的な sysedge.cf ファイル

インストーラによって導入されるファイルを示します。このファイルは ***Installed_Dir¥SystemEDGE¥config*** ディレクトリに置かれます。

デフォルト：

Windows：C:¥Program Files¥CA¥SystemEDGE¥config

UNIX/Linux：/opt/CA/SystemEDGE/config

動的な sysedge.cf ファイル

現行の **SystemEDGE** 設定ファイルを示します。このファイルは ***Data_Dir¥port<number>*** ディレクトリに置かれます。

デフォルト：

Windows：C:¥Users¥Public¥CA¥SystemEDGE¥port161

UNIX/Linux：/opt/CA/SystemEDGE/config/port161

オプション 1: エージェントをインストールして開始する前にバインドアドレスを設定する

エージェントがまだインストールされていない場合、バインドアドレスは以下のように設定できます。

1. エージェントを管理対象外モードでインストールします（マネージャ名は指定しません）。
2. エージェントが開始しないように、[インストール後に開始] オプションをオフにします。
3. インストール後に、静的な sysedge.cf ファイルを編集し、bind_address と manager_name を設定します。
4. 通常のメカニズムを使用してエージェントを開始します。

オプション 2: 管理されているシステムの前に実行中のシステムにバインドアドレスを設定する

管理対象外モードで SystemEDGE がインストールされている場合、バインドアドレスは以下のように設定できます。

1. 通常の方法で SystemEDGE を停止します。
2. 動的な `sysedge.cf` ファイルを編集し、`bind_address1` を設定します。
3. 必要に応じて、静的な `sysedge.cf` ファイルを編集し、`manager_name` を設定します。
4. 通常の方法を使用して SystemEDGE を開始します。

ここでの代替方法は、動的な `sysedge.cf` ファイルを削除し、代わりに静的な `sysedge.cf` ファイルで `bind_address` を設定することです。

オプション 3: 既存の管理対象システム上でバインドアドレスを設定する

SystemEDGE がポリシー設定によってすでに管理されている場合は、以下のようにバインドアドレスを設定します。

1. 通常の方法を使用して SystemEDGE を停止します。
2. 動的な `sysedge.cf` ファイルを削除します。
3. 動的な `sysedge.cf` ファイルと同じディレクトリにある `.sysedge.id` ファイルを削除します（これにより、エージェントは再度その設定をプッシュします）。
4. 必要に応じて、静的な `sysedge.cf` ファイルを編集し、`bind_address` を設定します。
5. 通常の方法を使用して SystemEDGE を開始します。

アクセスコミュニティの設定

インストール時に読み取り専用および読み取り/書き込み SNMPv1 アクセスコミュニティを定義し、`sysedge.cf` ファイルで追加のコミュニティを変更または定義できます。

注: SNMPv2c および SNMPv3 のアクセス情報は `sysedgeV3.cf` ファイルで定義します。

SNMPv1 アクセスコミュニティを設定するには、以下の形式を使用して `sysedge.cf` ファイルにアクセスコミュニティを入力するか、またはインストール時に指定されたエントリを編集します。

```
community <name> <access> <ip-address-list>
```

name

定義済みアドレスへの定義済みアクセスを許可するコミュニティを指定します。コミュニティ名として任意の **ASCII** 文字を使用できます。

access

付与する権限のレベル（読み取り専用または読み取り/書き込み）を指定します。

ip-address-list

指定されたコミュニティ名を使用してアクセスできる **IP** アドレスのスペース区切りリストを指定します。**IP** アドレスのスプーフィングが可能であるため、アクセスリストのすべてが安全だとは限りません。ただし、アクセスリストによって正当な使用を制限することは可能です。アクセスリストとして **IPv4** アドレスまたは **IPv6** アドレスを指定できます。

インストール後、**MIB** オブジェクトへの読み取り専用アクセスを提供する **public** という名前の読み取り専用コミュニティが **sysedge.cf** にデフォルトで 1 つ定義されます。

注: 慣例として、**public** というコミュニティ名を使用して読み取り専用アクセスを提供します。

MIB オブジェクトの値を変更するには、読み取り/書き込みアクセス権限があるコミュニティを定義する必要があります。以下の例では、**SystemEDGE** は **private** というコミュニティ名を使用して、**45.0.4.10**、**45.0.8.12**、**198.130.5.7**、**ea2f:fe90:abcd:0000:230:a2f:200:ad01** のいずれかの **IP** アドレスを持つシステムに読み取り/書き込みアクセスを許可します。**SystemEDGE** は **private** を使用しようとする他のすべてのシステムを認証失敗として扱います。

```
community private read-write 45.0.4.10 45.0.8.12 198.130.5.7
ea2f:fe90:abcd:0000:230:a2f:200:ad01
```

private というコミュニティ名は、ここでは例として使用しているだけです。セキュリティのため、より独特なコミュニティ名を使用してください。

注: **SNMP** メッセージヘッダに表示されるコミュニティ名は暗号化されていないため、アクセスリストが **IP** スプーフィングを受ける可能性があります。**SNMPv1** コミュニティを使用する場合は、潜在的なセキュリティ違反を制限する設定手順を実行する必要があります。詳細については、「セキュリティの設定に関する推奨事項」を参照してください。

SNMPv1 コミュニティのアクセス リスト

定義されたコミュニティのアクセス リストが空である場合、SystemEDGE は関連するコミュニティ名を使用するすべてのシステムにアクセスを許可します。アクセス リストには以下の制限が適用されます。

- `sysedge.cf` に指定されたアクセス リストは、SNMPv1 通信のみに使用されます。
- SNMPv2c および SNMPv3 のアクセス リストは、SNMPv3 の設定ファイル `sysedgeV3.cf` に別個に設定されます。SNMPv3 ユーザのアクセス リストを定義する方法の詳細については、「[SNMPv3 の設定](#) (P. 185)」を参照してください。
- IP アドレスはスペース文字で区切る必要があります。他の文字は改行を含めて使用できません。
- コミュニティ文字列ステートメント（アクセス リストを含む）の最大長は 1024 文字です。IPv4 アドレスを約 60 個指定できる十分なスペースがありますが、IPv6 アドレスの場合はそれより少なくなります。より長いアクセス リストを設定するには、別のコミュニティを定義してください。
- SystemEDGE エージェントには、`edgemon`、`edgemon`、および `emphistory` コマンドラインユーティリティが含まれます。これらのユーティリティは、エージェントに要求を送信するマネージャ システムとして機能します。エージェントがインストールされている同じシステム上でこれらのユーティリティを使用する場合は、アクセス リストにそのシステムの IP アドレスを追加します。

SNMPv1 トラップ ソースの指定

オプションで、SystemEDGE のトラップ PDU（protocol description unit）の送信元を示す IP アドレスである SNMPv1 トラップ ソースを指定できます。デフォルトでは、SystemEDGE は `gethostbyname` 関数呼び出しによって返された値を使用します。デフォルトの動作を無視するには、`trap_source` パラメータを指定します。このパラメータは、SNMPv1 トラップと SNMPv2c/SNMPv3 トラップの両方に適用されます。

エージェントにはトラップ ソースの値が 1 つしかないため、この値は `sysedge.cf` に 1 度だけ設定します。すべてのコミュニティ内のすべてのトラップが指定されたアドレスを使用します。指定するアドレスは有効な IPv4 アドレス、IPv6 アドレス、またはホスト名である必要がありますが、SystemEDGE エージェントはトラップ ソースとして使用される特定のシステムの有効なアドレスを指定したかどうかを判定するエラー チェックを行いません。

トラップソースを指定するには、`sysedge.cf` ファイルに以下の行を追加します。

```
trap_source <address>
```

address

トラップを送信するときに使用するソースアドレスを指定します。

IPv4 アドレス、IPv6 アドレス、またはホスト名を指定できます。

デフォルト：ホスト名

たとえば、トラップソースを IP アドレスが **10.0.7.73** でホスト名が **system1.ca.com** のシステムに設定するには、以下のどちらかを入力します。

```
trap_source 10.0.7.73
```

```
trap_source system1.ca.com
```

SNMPv1 トラップ デスティネーションの設定

`sysedge.cf` ファイルには、SystemEDGE エージェントに SNMPv1 トラップメッセージの送信先を指示する SNMPv1 トラップ デスティネーションの定義が含まれています。任意の数の管理システムにトラップを送信するようにエージェントを設定できます。SNMPv1 トラップ デスティネーションは、インストール時に、または `sysedge.cf` ファイルで定義できます。

SNMPv1 トラップ デスティネーションを設定するには、SNMPv1 トラップの送信先となるシステムごとに `sysedge.cf` ファイルに以下の行を追加します。

```
trap_community <name> <address> [<port> [<encoding> [<source>]]]
```

name

トラップで送信されるコミュニティ名を指定します。

address

トラップを送信する **IPv4 アドレス**または **IPv6 アドレス**を指定します。

port

(オプション) トラップの送信先となる **UDP ポート**を指定します。

デフォルト： **162**

encoding

(オプション) **trap_source** パラメータで定義したソース アドレスをトラップに含める方法を指定します。このパラメータは、**trap_source** が IPv6 アドレスに変換される場合に重要です。3 桁の XYZ の形式（先頭の未使用桁をゼロで埋める）でエンコーディング パラメータを入力します。

デフォルト： 000

X

4 バイトの IPv4 ソース アドレス フィールド (SNMPv1 トラップのみ) の拡張を制御します。16 バイトの IPv6 アドレスを含めるのに、ソース アドレス フィールドを拡張しない場合は 0 を入力し、ソース アドレス フィールドを拡張する場合は 1 を入力します。

Y、Z

トラップの **varbind** (Y) または UDP パケット (Z、SNMPv1 トラップのみ) にソース情報を含めるかどうかを制御します。これらの桁に以下のいずれかを入力します。

0：トラップの **varbind** または外部 UDP パケットは変更されません。

1： **varbind** またはパケットに **trap_source** パラメータ (IPv4/IPv6 アドレスまたはホスト名) をそのまま含めます。

2： **trap_source** パラメータを、可能であれば IPv4 アドレスとして（続いて IPv6 アドレス、ホスト名の順に試行）含めます。

3： **trap_source** パラメータを、可能であれば IPv6 アドレスとして（続いて IPv4 アドレス、ホスト名の順に試行）含めます。

4： **trap_source** パラメータを、可能であればホスト名として（続いて IPv4、IPv6 の順に試行）含めます。

5： 2 の優先順位に従い、ホスト名を含めます。

6： 3 の優先順位に従い、ホスト名を含めます。

7： 1 の優先順位に従い、ホスト名を含めます (**trap_source** が IPv6 アドレスの場合)。

ソース

トラップソースとして使用する **IPv4 アドレス**、**IPv6 アドレス**、または **ホスト名**を指定します。

デフォルト: **trap_source** パラメータの値

たとえば、**trap_source** パラメータとホスト名を **UDP** パケットに付加した **mycommunity** というコミュニティ名のトラップを **IP アドレス 10.16.5.26** のポート **1692** に送信するには、以下の行を追加します。

```
trap_community mycommunity 10.16.5.26 1692 7
```

注: **sysedge.cf** ファイルは **SNMPv1** トラップデスティネーションのみを定義します。 **SNMPv2c** および **SNMPv3** トラップを設定する方法の詳細については、「**SNMPv3 の設定**」を参照してください。

SystemEDGE から送信されるトラップのエージェントアドレス

SystemEDGE エージェントから送信されるトラップのソースアドレスは、エージェントがバインドされているアドレスです。デフォルトでは、**SystemEDGE** エージェントはすべてのネットワーク インターフェースにバインドされているため、エージェントから送信されるトラップは最初に送信に成功した **IP アドレス** を使用します。

エージェントと同じローカルサーバ上で実行されているトラップレシーバ (**edgetrapmon** など) にトラップを送信するように **SystemEDGE** を設定すると、ソースアドレスはほとんどの場合ループバックアドレス (**IPv4** では **127.0.0.1**、**IPv6** では **::1**) になります。

認証失敗トラップの設定

コミュニティ名がエージェントによって認識されているコミュニティと一致しない **SNMP** メッセージを受信したときは常に認証失敗トラップを送信するように **SystemEDGE** エージェントを設定できます。

デフォルトでは、エージェントは認証失敗トラップを送信しません。認証失敗トラップを送信するようにエージェントを設定するには、以下のように **sysedge.cf** 内の **no_authen_traps** ディレクティブの行頭にポンド記号 (**#**) 文字を付けて、この行をコメントアウトします。

```
# no_authen_traps
```

注: このオプションは、**SNMPv1** トラップ (**sysedge.cf** に設定) および **SNMPv2c/SNMPv3** トラップ (**sysedgeV3.cf** に設定) に対して設定されたトラップデスティネーションの認証失敗トラップに適用されます。

SNMP SET 通知の設定

CA Virtual Assurance で SystemEDGE エージェントを管理すると、エージェントの設定に対して **SNMP SET** の変更が実行されたときに **CA Virtual Assurance** マネージャがデフォルトで通知を受信します。これらの通知により、**CA Virtual Assurance** マネージャはエージェント設定に対するすべての変更を認識し、不要な変更をロールバックできるようになります。

注: この機能は、**CA Virtual Assurance** と共に **SystemEDGE** エージェントを使用する場合に適用されます。これらの通知を設定する方法の詳細については、「**CA Virtual Assurance** 管理ガイド」を参照してください。

この機能は、**CA Virtual Assurance** でエージェントを管理しているときにデフォルトで有効になります。**CA Virtual Assurance** マネージャに対する **SNMP SET** 通知を無効にするには、**sysedge.cf** ファイルに以下の行を追加します。

```
no_snmp_set_notifications
```

SNMP SET 制限の設定

デフォルトでは、SystemEDGE エージェントは Systems Management Empire MIB および Host Resource MIB 内の以下のテーブルに関する SNMP SET 要求を許可しません。

リモートシェルグループ

有効にすると、管理システムはエージェントシステム上でシェルスクリプトとプログラムを実行するようにリモートからエージェントに指示できます。このタイプの情報の漏えいは、潜在的なセキュリティリスクになる可能性があります。

注: このグループの詳細については、「Systems Management Empire MIB」の章を参照してください。

プロセステーブルと実行中のソフトウェアテーブル

エージェントシステム上で実行されているプロセスおよびその他のソフトウェアへのアクセスを許可します。これらのテーブルに関する **SNMP Set** 要求を許可するときは、セキュリティリスクを考慮してください。

注: プロセステーブルの詳細については、「Systems Management Empire MIB」の章を参照してください。実行中のソフトウェアテーブルの詳細については、付録「Host Resources MIB」を参照してください。

これらのテーブルに対する **SNMP Set** を有効にすることができますが、その場合は SNMPv1 および SNMPv2c 通信用の読み取り/書き込み権限を持つ有効なコミュニティまたは SNMPv3 通信用の有効な SNMPv3 ユーザ認証情報をクエリに使用する必要があります。

リモートシェルグループに対する **SNMP Set** のサポートを有効にするには、`sysedge.cf` ファイルの以下の行にポンド記号 (#) 文字を追加して、この行をコメントアウトします。

```
# no_remoteshell_group
```

プロセステーブルおよび実行中のソフトウェアテーブルに対する **SNMP Set** のサポートを有効にするには、`sysedge.cf` ファイルの以下の行にポンド記号 (#) 文字を追加して、この行をコメントアウトします。

```
# no_process_sets
```

これらの行を完全に削除することもできます。

GetBulk 応答メッセージ サイズの設定

GetBulk 要求の応答サイズを制限するように SystemEDGE エージェントを設定できます。GetBulk 応答メッセージペイロードのバイトサイズを制限すると、SystemEDGE エージェントが使用するメモリも制限されます。

応答メッセージが制限を超えると、SystemEDGE エージェントは有効な応答メッセージの形で要求された OID のリストを繰り返し提供します。たとえば、IP パケットのフラグメンテーションを回避するため、ソフト制限を 60000 に設定できます。IP パケットの最大サイズは 65535 であるため、これでもオーバーフロー制限には達しません。

GetBulk 応答を制限するようにエージェントを設定するには、以下のように `sysedge.cf` に `bulk_read_response_soft_limit` オプションを追加します。

```
bulk_read_response_soft_limit <limit>
```

ここで、`limit` には応答メッセージの制限として設定するバイト数を指定します。符号なし整数で、デフォルト値は 60000 です。

注: このオプションは、SNMPv2c/SNMPv3 トラップ (`sysedge.cf` および `sysedgeV3.cf` に設定) に対して設定されたトラップデスティネーションの認証失敗トラップに適用されます。

連邦情報処理規格(FIPS)モードの設定

`sysedge_fips_mode` パラメータを使用して、SystemEDGE エージェントの暗号化処理方法を設定できます。以下の 3 つの暗号化方式が利用可能です。

- 最小限のセキュリティ機能 (MD5/SHA および DES のみ) を保証する内部ソリューション
- CA eTrust 公開鍵インフラストラクチャ ライブラリ (libetpki/libcapki)
- RSA BSAFE Crypto-C Micro Edition ライブラリ (libcryptocme2) (FIPS 140-2 準拠)

暗号化モードを設定するには、以下のように `sysedge.cf` ファイルに `sysedge_fips_mode` パラメータを含む行を追加します。

```
sysedge_fips_mode <method>
```

method

使用する暗号化のタイプを指定します。以下のオプションを使用できます。

0

CA eTrust 公開鍵インフラストラクチャ ライブラリが有効になります。この方式が失敗する場合は、最小限のセキュリティを提供する内部ソリューションに戻ります。

1

FIPS 準拠の暗号化が有効になります。この方式が失敗する場合は、方式 0 に戻ります。これは、パラメータを設定しなかった場合のデフォルトです。

2

エージェントが **FIPS** のみのモードで動作することを指定します。
RSA BSAFE Crypto-C Micro Edition FIPS 準拠ライブラリが有効になります。失敗する場合は、暗号化を実行しません。

たとえば、FIPS 認定プロトコルと FIPS 認定ライブラリのみを使用してエージェントを実行するには、以下の行を入力します。

```
sysedge_fips_mode 2
```

FIPS 暗号化を有効にする方法の詳細については、付録「FIPS 140-2 の暗号化」を参照してください。

診断ロギングメカニズムの設定

SystemEDGE エージェントは、CA サポートが主に使用するファイルに診断メッセージを記録するための共通のロギングメカニズムを備えています。ロギングメカニズムの以下の項目を設定できます。

- ログレベル
- ログファイルの名前、サイズ、およびサイズ制限の処理メカニズム
- どのモジュールのメッセージをログに記録するか

エージェントのログ記録を設定するには、`sysedge.cf` ファイルに以下の行を入力します。

```
sysedge_loglevel <level>
sysedge_logfile '<file>' [<size> [<number>]]
sysedge_logpattern <modules>
```

level

ログに記録するメッセージのレベルを指定します。以下のいずれかを指定できます。

- fatal
- critical
- warning
- info
- debug
- debug[1-3]

エージェントは、指定したレベルと同じまたはそれ以上の重大度を持つメッセージをログに記録します。たとえば、**warning** を指定すると、エージェントはすべての警告、重大、および致命的メッセージをログに記録します。

デフォルト: **info**

'file'

ログファイルの名前を指定します。ファイル名を含む絶対パス名を指定できます。絶対パス名を指定しなかった場合、ログファイルはエージェントのポート固有のデータディレクトリに作成されます。

デフォルト: **'sysedge.log'**

サイズ

ファイルの最大サイズ (KB 単位) を指定します。このパラメータをゼロ (0) 以下の値に設定した場合は、ログファイルのサイズが制限されません。

デフォルト : 10240

数値

サイズ制限に達したときのログファイルの処理を制御します。以下のいずれかのオプションを選択できます。

0

サイズ制限に達したときにログ記録を停止します。

1

サイズ制限に達したときに現在のログファイルの内容を「<file>.1」に保存し、ログ記録をログファイルの先頭に折り返します。

>1

指定された数までのバックアップファイルから成る循環セット (またはリング) をログ記録に使用します。バックアップファイルには、「<file>.<number>」という名前が付けられ、常に最も古いログエントリが格納されます。

-1

サイズ制限に達したときにログ記録をログファイルの先頭に折り返します。

-2

サイズ制限に達したときに現在のログファイルの内容を「<file>.<YYYY-MM-DD_hh.mm.ss>」に保存し、ログ記録をログファイルの先頭に折り返します。

デフォルト : 1

modules

ログに記録するメッセージの生成元モジュールのスペース区切りリストを指定します。指定できるモジュールは、以下のとおりです。

- **se** : SystemEDGE エージェント
- **svcrsp** : サービス レスポンス モニタ AIM
- **rmonwbem** : リモート モニタリング AIM
- **monwin** : モニタ保守ウィンドウ AIM
- **perfcube** : パフォーマンス キューブ AIM
- **util** : 共通/共有ライブラリ
- **regexpr** : 共通/共有ライブラリ
- **shm** : 共通/共有ライブラリ
- **デフォルト** : すべてのモジュールのログ メッセージ

以下の例では、SystemEDGE エージェントおよび SRM AIM に関する重大度が「情報」以上のすべてのメッセージを **sysedgemsg.log** という名前のファイルに記録し、30000 KB の制限と古いファイルをすべて保存する折り返し式サイズ制限メカニズムを設定します。

```
sysedge_loglevel info
sysedge_logfile sysedgemsg.log 30000 -2
sysedge_logpattern se svcrsp
```

保守モードの設定

SystemEDGE エージェントは、すべてのモニタ エントリの処理とトラップの送信が停止される保守モードで実行できます。保守モードは、エージェントのシステムの計画休止を実施するときに、虚偽のアラーム トラップの受信を回避したい場合に有用です。

保守モードでは、エージェントはメトリックの収集と SNMP 要求への応答を続行しますが、すべてのモニタと履歴の収集の処理を中断します。エージェントは、外部に対してすべてのモニタの行ステータスを **notInService** としてレポートしますが、内部では正しい行ステータスを維持します。エージェントは、保守ウィンドウの開始時にすべてのモニタの現在値を保存し、保守ウィンドウの終了時にそれを現在値と比較して、必要な場合は現在値に対応するトラップを送信します。

保守モードを有効にするには、`sysedge.cf` ファイルに以下の行を追加します。

```
sysedge_maintenance
```

設定ファイルで保守モードを管理するより、**CA Virtual Assurance** マネージャ インターフェースの保守モード制御を使用する方が効率的です。インターフェースから保守モードを切り替えることができ、設定を変更すると自動的に設定ファイルが更新され、展開されます。**CA Virtual Assurance** を使用して設定すると、エージェントは完全な再起動ではなくウォーム スタートを実行します。**CA Virtual Assurance** で保守モードを有効にする方法の詳細については、「**CA Virtual Assurance** 管理ガイド」およびオンライン ヘルプを参照してください。

アクションのサポートの設定

デフォルトでは、**SystemEDGE** エージェントはモニタリングテーブルでのアクション コマンドの実行を許可しません。アクション コマンドやアクション スクリプトから `root` またはユーザ管理者としてコマンドを実行できるため、アクション コマンドやアクション スクリプトを実行する機能は潜在的なセキュリティ問題になります。特定のモニタリングしきい値に達したときにアクションを実行する場合は、アクションのサポートを有効にする必要があります。

アクション実行のサポートを有効にするには、`sysedge.cf` ファイルの以下の行にポンド記号 (`#`) 文字を追加して、この行をコメントアウトします。

```
# no_actions
```

サブプログラムのユーザおよびグループ権限の設定 (UNIX のみ)

デフォルトでは、**SystemEDGE** エージェントは有効なユーザおよびグループ権限（通常は `root`）でサブプログラム（たとえば、リモート シェル、アクション、拡張オブジェクト呼び出しなど）を実行します。適用されるローカル セキュリティ ポリシーによっては、別のユーザおよびグループ権限で実行されるアクションや拡張オブジェクトを使用するようにエージェントを設定したい場合があります。

`root` 以外の有効なユーザおよびグループ権限でサブプログラムを実行するには、`sysedge.cf` ファイルに以下の行を追加します。

```
subprogram_user_name <name>  
subprogram_group_name <name>
```

name

サブプログラムに使用するユーザ名またはグループ名を指定します。指定した名前が無効な場合は、すべてのサブプログラム実行が無効になります。

MIB テーブルの制限の設定

`sysedge.cf` ファイルには、SystemEDGE エージェントが Systems Management Empire MIB 内の特定のテーブルにデータを入力しないようにするためのパラメータがいくつか存在します。どのパラメータもデフォルトでは入力されていませんが、これらを追加することにより、機密情報が含まれるテーブルや大量のリソースを消費するテーブルへのアクセスを制限できます。

特定の MIB テーブルのサポートを無効にするには、制限するテーブルに応じて `sysedge.cf` ファイルに以下の行を入力します。

`no_process_table`

プロセス テーブルへの入力を制限します。このテーブルは基盤となるシステム上で実行されているプロセスを検出できますが、これは適用されているローカルセキュリティ ポリシーに違反する可能性があります。

`no_topprocs_table`

トップ プロセス テーブルへのオブジェクトの入力を制限します。

`no_who_table`

システムに現在ログインしているユーザに関する情報を提供する **Who** テーブルへの入力を制限します。このタイプの情報の漏えいは、潜在的なセキュリティ リスクになる可能性があります。

`no_usergroup_table`

システムに設定されたユーザ アカウントおよびユーザ グループに関する情報を提供するユーザ テーブルおよびグループ テーブルへの入力を制限します。組織がユーザおよびグループの情報の漏えいをセキュリティの問題と見なしている場合や、数多くのユーザおよびグループを含む分散システムが存在する場合は、これらのテーブルへのアクセスを制限できます。エージェントはこれらの情報を内部で定期的にキャッシュするため、ユーザおよびグループの情報を格納すると大量のリソースが消費される可能性があります。

`no_trapcommunity_table`

エージェントの現在の **SNMP** 設定に関する情報（バージョン、コミュニティ、トラップ デスティネーションを含む）を提供するトラップ コミュニティ テーブルへの入力を制限します。コミュニティなどの **SNMP** 設定は機密性が高いため、このテーブルへのアクセスを制限したい場合があります。

no_mirror_monitor_table

セルフ モニタ テーブルの読み取り専用ミラーを提供するミラー モニタ テーブルへの入力を制限します。このテーブルを使用すると、セルフ モニタ テーブル全体を検索せずに特別なコンテキストでモニタ エントリを表示できます。セルフ モニタ テーブルのミラーが消費するリソースが多すぎる場合は、このテーブルを無効にすることができます。

no_mirror_aggregate_table

集計テーブルの読み取り専用ミラーを提供するミラー集計テーブルへの入力を制限します。このテーブルを使用すると、集計テーブル全体を検索せずに特別なコンテキストで集計エントリを表示できます。集計テーブルのミラーが消費するリソースが多すぎる場合は、このテーブルを無効にすることができます。

注: これらのテーブルの詳細については、「Systems Management Empire MIB」の章を参照してください。

デバイス ステータス確認の制限の設定

sysedge.cf ファイルには、一般にエージェントがブロックされる可能性を回避するため、SystemEDGE エージェントが特定のシステム デバイスのステータスを照会しないようにするためのパラメータがいくつか存在します。デフォルトでは、SystemEDGE エージェントはシステム上のすべてのフロッピー デバイスのステータスを自動的に判定しません。

フロッピー ドライブのステータス確認を有効にするには、以下のように sysedge.cf ファイルの以下の行にポンド記号 (#) 文字を追加して、この行をコメントアウトします。

```
# no_stat_floppy
```

この行を完全に削除することもできます。

デフォルトで有効になっている他のデバイス ステータス確認を無効にすることができます。デバイス ステータス確認を無効にするには、制限するデバイスに応じて `sysedge.cf` ファイルに以下の行を入力します。

注: これらの制限を有効にすると、それぞれのオブジェクトに関して一部の情報が提供されなくなるため、対応する MIB テーブルに影響します。

`no_serial_status`

ホスト システム上のすべてのシリアル デバイスのステータス確認を無効にします。エージェントは、**Host Resources MIB** の `hrDeviceTable` に対するサポートの一部として、デフォルトでこの確認を行います。この操作によって、一部のシリアルアプリケーションは他のプロセスによるシリアル デバイスのオープンとクローズ (ステータスの判定に必要) を処理できなくなり、問題が発生します。この行を入力すると、エージェントは `hrDeviceTable` のキーボードとマウスのみを確認し、管理システムのクエリに対してシリアル ポートのステータスとして不明 (1) を返します。

`no_discover_disks`

SystemEDGE 上でディスク デバイスの検索および検出オプションを有効にします。このオプションが無効な場合、ディスク デバイスは以下の **SystemEDGE SNMP** テーブルに表示されません。

- `hrStorageTable`
- `hrDeviceTable`
- `hrDiskStorageTable`
- `hrFSTable`
- `devTable`
- `diskStatsTable`

注: `no_discover_disks` オプションが有効な場合、`no_probe_disks` オプションは自動的に有効になります。

`no_probe_disks`

古い **UNIX** システム上でドライバがステータス情報を待機する間にエージェントが長時間ブロックされる可能性を回避するため、基盤となるシステムにインストールされたディスクおよび **CD-ROM** のサイズ、容量、説明、その他のプロパティの確認を無効にします。

注: この行を追加すると、エージェントはディスク デバイスを検出しますが、ディスクの統計情報、容量情報、デバイスの説明、およびステータス情報を提供できなくなる可能性があります。

`no_stat_nfs_filesystems`

(UNIX のみ) エージェントが **Systems Management Empire MIB** を介してデフォルトで行うローカルおよびリモート ファイル システム データの公開を無効にします。一部の **UNIX** システムでは、利用できなくなったリモート ファイル サーバからファイル システムがマウントされた場合に、ファイル システム データのチェックによってエージェントがブロックされる可能性があります。この状況でエージェントのブロックを解除する方法はありません。

注: この行を追加すると、NFS ファイル システムだけでなく、すべてのリモート ファイル システムの確認が無効になります。

`no_hpux_probe_graphics`

エージェントがブロックされる可能性を回避するため、**HP-UX** システム上のグラフィックスのステータス確認を無効にします。

ウォーム スタート ディスカバリの設定

CA Virtual Assurance で SystemEDGE エージェントを管理している場合は、CA Virtual Assurance マネージャを使用してエージェントの設定を更新すると、エージェントがウォーム スタートを実行します。エージェントは完全な再起動を実行しなくても、ウォーム スタートによって変更を実装できます。デフォルトでは、エージェントのウォーム スタートが発生しても、数多くのデバイスを含むシステムを管理している場合はデバイスの検出に時間がかかるため、エージェントは必ずしもすべてのデバイスを再検出しません。しかし、このディスカバリを有効にすることで、ウォーム スタート後にすべてのデバイスを再検出できます。

ウォーム スタート ディスカバリを有効にするには、以下のように `sysedge.cf` ファイルの以下の行にポンド記号 (#) 文字を追加して、この行をコメントアウトします。

```
# no_warmstart_discovery
```

ウォーム スタート機能は、CA Virtual Assurance の管理対象モードのエージェントにのみ適用されます。詳細については、「CA Virtual Assurance 管理ガイド」を参照してください。

Linux の空きメモリ計算の設定

SystemEDGE が Linux 上の空きメモリを計算するときに、システム バッファ、キャッシュ メモリ、またはその両方を計算に含めるかどうかを設定できます。

Linux の空きメモリ計算を設定するには、`sysedge.cf` ファイルに以下の行を入力します。

```
linux_freemem_include [buffers] [cached]
```

buffers

システム バッファを空きメモリ計算に含めます。

cached

ディスク キャッシュ メモリを空きメモリ計算に含めます。

たとえば、システム バッファとディスク キャッシュ メモリの両方を空きメモリ計算に含めるには、以下の行を追加します。

```
linux_freemem_include buffers cached
```

多層階層でのモニタ集計の設定

SystemEDGE は、セルフ モニタとプロセス モニタのエントリを集計し、同じ属性の複数のモニタを相関させて、最も高い重大度のみを表示できます。集計されたモニタは、同じオブジェクトクラス、インスタンス、および属性を持っています。デフォルトでは、エージェントは集計モニタのエントリを管理するときに、トラップを送信し、集計された最悪の状態を持つエントリに対してのみアクションを実行します。このデフォルト設定を `sysedge.cf` ファイルで変更できます。

トラップを送信し、最も高い重大度だけでなく、すべての接続済みモニタに対してアクションを実行するには、`sysedge.cf` に以下の行を追加します。

```
aggregate_monitor_traps  
aggregate_monitor_actions
```

モニタ集計のレベルを設定することもできます。デフォルトでは、エージェントは同じオブジェクトクラス、インスタンス、および属性を持つすべてのセルフモニタおよびプロセスモニタを標準集計に集計します。ある集計に属するすべてのモニタは、*接続済みモニタ*と呼ばれます。sysedge.cf ファイルでは、より高いレベルの集計を設定できます。より高いレベルの集計をスーパー集計と呼びます（プレースホルダとしてアスタリスク「*」を使用します）。たとえば、オブジェクト集計テーブルで以下のようなモニタ集計を作成できます。

- 同じオブジェクトクラス、同じ属性、および任意のインスタンスによるモニタ
- 同じクラス、任意のインスタンス、および任意の属性によるモニタ

仮想オブジェクトインスタンスの集計は3つ以上の層をサポートします。サポートされているインスタンスの構造は、以下のとおりです。

```
//<any1>/<any2>/.../<anyN>/<instance>
```

例

LPAR 上のオブジェクト インスタンス :

```
//hmc1/hyper1/lpar1/disk1
```

スーパー集計は、すべての層ではなく2つの層（右端の2つの層）にのみ提供されます。この例では、以下のスーパー集計が作成されます。

```
//hmc1/hyper1/lpar1/*
```

```
//hmc1/hyper1/*/disk1
```

```
//hmc1/hyper1/*/*
```

すべてのインスタンスを集計するため、特別なスーパー集計 `//*` が作成されます。

注: オブジェクトインスタンスでは、「`//hierarchy/instance`」のように「`//`」と「`/`」を区切り文字として使用して、インスタンスとそれをホストする階層とを関連付けます。階層は多層になる場合があります。ローカルシステムインスタンスについては、「`///.`」を先頭に付加します。

例

```
//VM-Host/VM-Guest/Disk1
```

```
//Host/Disk1
```

```
///./Disk1
```


スーパー集計は、右端の 2 つの層の一方または両方で有効にできます。すべてのインスタンスを集計するため、特別なスーパー集計 `//*` が作成されます。

レベル 0002 の例

```
//VM-Host/VM-Guest/*
//Host/*
//.*/*
```

レベル 0004 の例

```
//VM-Host/*/Disk1
//*/Disk1          (//./Disk1 を除く)
```

レベル 0008 の例

```
//VM-Host/*//*
//*//*             (//./Disk1 を除く)
//*
```

モニタ エントリの集計レベルを設定するには、`sysedge.cf` ファイルに以下の行を追加します。

```
aggregate_level <level>
```

```
level
```

以下のテーブルの集計レベルを許可するビット フィールドを指定します。

クラス	インスタンス	属性	レベル
クラス	<code>//hierarchy/instance</code>	属性	(標準集計)
クラス	<code>//hierarchy/*</code>	属性	0002
クラス	<code>//hierarchy-1/*/instance</code>	属性	0004
クラス	<code>//hierarchy-1/*//*</code>	属性	0008
クラス	<code>//hierarchy/instance</code>	*	0010
クラス	<code>//hierarchy/*</code>	*	0020
クラス	<code>//hierarchy-1/*/instance</code>	*	0040
クラス	<code>//hierarchy-1/*//*</code>	*	0080
*	<code>//hierarchy/instance</code>	属性	0100
*	<code>//hierarchy/*</code>	属性	0200
*	<code>//hierarchy-1/*/instance</code>	属性	0400
*	<code>//hierarchy-1/*//*</code>	属性	0800

クラス	インスタンス	属性	レベル
*	//hierarchy/instance	*	1000
*	//hierarchy/*	*	2000
*	//hierarchy-1/*/instance	*	4000
*	//hierarchy-1/*/*	*	8000

たとえば、`sysedge.cf` ファイルに以下の行を追加すると、すべての集計ルールが適用され、エージェント全体の状態が集計対象になります。

```
aggregate_level 0xffff
```

モニタ集計およびセルフ モニタ テーブルのオブジェクト クラス列、インスタンス列、属性列の詳細については、「概念」および「セルフ モニタリング」の章を参照してください。

モニタリング設定

`sysedge.cf` ファイルには、SystemEDGE エージェントのモニタリング操作を制御するモニタ エントリが格納されています。モニタ エントリは以下の方法で作成できます。

- CA Virtual Assurance のユーザ インターフェースで動的に作成する方法（再起動せずに任意のエージェント システムに変更を展開できる）
- リモートまたはローカルで **SNMP Set** 要求を使用して作成する方法
- ローカルの `sysedge.cf` ファイルを直接編集し、再起動後に実装する方法

`sysedge.cf` ファイルに行を追加することにより、以下の **MIB** テーブルにモニタリング エントリを追加できます。

- ファイル システム、インターフェース、プロセッサなどの **MIB** オブジェクトのしきい値をモニタするためのセルフ モニタ テーブル。セルフ モニタ テーブルは、**MIB** 属性の **OID** またはオブジェクトのクラス、インスタンス、および属性値に基づくモニタリング エントリの入力と属性しきい値の設定をサポートします。
- プロセス属性と **Windows** サービスをモニタするためのプロセス モニタ テーブル。プロセス モニタ テーブルは、プロセスまたはサービスの正規表現に基づくモニタリング エントリの入力とプロセス属性しきい値の設定をサポートします。
- プロセス グループを定義してモニタするためのプロセス グループ モニタ テーブル。プロセス グループ モニタ テーブルは、プロセスの正規表現に基づいてモニタリング エントリを入力し、変更やグループ統計をモニタするグループを作成する作業をサポートします。
- 正規表現パターンと一致するログ ファイルをモニタするためのログ モニタ テーブル。ログ モニタ テーブルは、ログ ファイルおよびログ ファイル内の正規表現パターンに基づくエントリの入力をサポートします。

- 正規表現パターンと一致する Windows イベント ログをモニタするための NT イベント モニタ テーブル。NT イベント モニタ テーブルは、Windows イベント ログ内で照合する正規表現パターンに基づくエントリの入力をサポートします。
- ベースライン設定や傾向分析に使用する履歴データを収集するための履歴制御テーブル。履歴制御テーブルは、整数ベースの MIB 属性の OID に基づくエントリの入力をサポートします。

エージェントの状態管理モデルでは、セルフ モニタ テーブルとプロセス モニタ テーブルのエントリから状態と重大度の情報を抽出し、接続済みモニタを集計して重大度の最も高いステータスのみを表示および処理します。

これらのモニタリング テーブルの動作方法の詳細については、「概念」の章を参照してください。これらのテーブルの列および `sysedge.cf` ファイルに各テーブル用のエントリを作成する方法の詳細については、個々のモニタリング タイプの章を参照してください。CA Virtual Assurance のユーザ インターフェースからモニタ エントリを作成する方法の詳細については、「CA Virtual Assurance 管理ガイド」およびオンライン ヘルプを参照してください。

注: `sysedge.cf` ファイルには、モニタする MIB 拡張や Windows レジストリまたはパフォーマンスのデータを定義することもできます。詳細については、「カスタム MIB オブジェクト」および「Windows レジストリとパフォーマンス MIB オブジェクト」の章を参照してください。

SystemEDGE AIM のロード

SystemEDGE エージェントは、初期化時にオプションの SystemEDGE AIM をロードできるプラグインアーキテクチャを提供します。これらの AIM により、拡張性の高い柔軟な方法でアプリケーション固有の MIB 変数をサポートできます。

`sysedge.cf` ファイルを編集して、エージェントがロードする AIM を指定できます。SystemEDGE エージェントがロードする AIM を検索するための絶対パスまたは AIM 名のみを指定します。

すべてのプラットフォーム上の SystemEDGE ディストリビューションには、以下のオンデマンドモジュールが含まれています。これらのモジュールは、AIM と異なり、機能が必要になったときに暗黙的にロードされます。

- モニタ保守ウィンドウ
- パフォーマンス キューブ

注: トップ プロセス AIM に以前含まれていた機能は、ベース エージェントに統合されました。詳細については、「プロセスとサービスのモニタリング」の章を参照してください。

SystemEDGE AIM をロードするには、`sysedge.cf` ファイルの以下の行に AIM 名または AIM の名前とフルパスを入力します。

```
sysedge_plugin
```

たとえば、SRM AIM をロードするには、`sysedge.cf` ファイルに以下を入力します。

```
sysedge_plugin svcrsp
```

絶対パスを入力する必要はありません（ただし、絶対パスを入力しても有効なエントリが作成されます）。SystemEDGE エージェントは、AIM をロードするためのプラットフォームとファイルの場所を自動的に抽出します。

他の AIM を有効にする方法の詳細については、その AIM のドキュメントを参照してください。AIM の設定の詳細については、「SystemEDGE AIM」の章を参照してください。

モニタ保守ウィンドウ モジュールの設定

モニタ保守ウィンドウ モジュールは、SystemEDGE のモニタリングを時間ベースで制御する機能を備えています。モニタ保守ウィンドウ モジュールは、一日の重要でない時間帯にモニタ自体を停止することなく、セルフ モニタ、プロセス モニタ、プロセス グループ モニタ、ログ ファイル モニタ、履歴モニタ、および Windows イベント モニタからのトラップの生成を停止できます。

このオンデマンドのモジュールは、SystemEDGE エージェントに付属しており、`sysedge.cf` ファイルでその操作を設定できます。このモジュールを設定するための行をファイルに追加すると、その設定は自動的に有効になります。

このモジュールの詳細については、「SystemEDGE AIM」の章を参照してください。

Perfcube モジュールの設定

Perfcube モジュールは、CA NSM Systems Performance によるパフォーマンス分析に使用する Systems Performance キューブ内の履歴テーブルのメトリックを収集します。このモジュールは、履歴テーブル内のエントリを読み取り、定義された間隔でメトリックを平均して、キューブ ファイルにメトリックの平均値を書き込みます。

このオンデマンドのモジュールは、SystemEDGE エージェントに付属しており、sysedge.cf ファイルでその操作を設定できます。以下のパラメータを使用してファイルに 0 より大きいキューブ収集間隔を設定すると、AIM は自動的に有効になります。

```
emphistory cube_interval interval  
interval
```

履歴情報をキューブ ファイルに収集する間隔を分単位で指定します。

Perfcube モジュールの設定の詳細については、「SystemEDGE AIM」の章を参照してください。

セキュリティの設定に関する推奨事項

セキュリティのため、以下の設定パラメータを実装することをお勧めします。

```
no_who_table  
no_usergroup_table  
no_remoteshell_group  
no_process_sets
```

セキュリティの実装に役立つ追加設定オプションを以下に示します。

```
no_actions  
no_process_table  
no_trap_community_table  
no_mirror_monitor_table  
no_mirror_aggregate_table  
subprogram_user_name  
subprogram_group_name
```

モニタおよび自動ウォッチャーの正規表現の例

このセクションでは、各モニタ タイプに応じた正規表現の例を示します。

プロセス モニタ

- 「svchost」で始まるプロセスに一致します。

`^svchost`

- 「sql」が含まれるプロセスに一致します。

`sql`

- 「svc」で終わるプロセスに一致します。

`svc$`

- 「cmd」および「CMD」両方のプロセスに一致します（PCRE が必要です）。

`/cmd/i`

注: プロセス（0x4000 フラグ付き）の複数のインスタンスが存在することをモニタしていない場合、1つのプロセスにのみ一致する正規表現を使用します。

Windows サービス モニタ

- 正確な表示名が「Print Spooler」であるサービスに一致します。

`^Print Spooler$`

注: Windows サービスについては、1つのサービスにのみ一致する正規表現を使用します。

プロセス グループ モニタ

- 「sql」または「svchost」で始まるプロセスすべてに一致します。

`^sql|^svchost`

Windows イベント モニタ(ソースフィルタ)

- 任意のソースからのイベントに一致します。

`.*`

- http サーバからのイベントに一致します。

`http`

Windows イベント モニタ(説明フィルタ)

- イベント ID が 277 のイベントに一致します（「イベント ID を先頭に追加」フラグ - 0x0100 が必要です）。

¥[277¥]

- 行内の任意の場所にテキスト「FATAL」が含まれるイベントに一致します。

FATAL

ログ ファイル モニタ(検索フィルタ)

- su の試行失敗を示すテキストに一致します。

su.*fail

- ログ ファイル内のいくつかの行によって区切られたテキスト文字列「WARNING:」および「Invalid login attempt」に一致します (PCRE が必要です)。

/^WARNING:(.*¥n)*Invalid login attempt/m

- 大文字小文字を区別せずにテキスト「error」に一致します (PCRE が必要です)。

/error/i

汎用自動ウォッチャー

インスタンス条件正規表現、正規表現 2、正規表現 3 :

- 「WiFi」が含まれるテキストに一致します。

WiFi

- 「C:」、「D:」、または「E:」が含まれるテキストに一致します。

C:|D:|E:

プロセス自動ウォッチャー

インスタンス条件正規表現、正規表現 2、正規表現 3 :

- 「svchost」で始まるプロセスに一致します。
^svchost
- 「sql」が含まれるプロセスに一致します。
sql
- 「svc」で終わるプロセスに一致します。
svc\$
- 「cmd」および「CMD」両方のプロセスに一致します（PCRE が必要です）。
/cmd/i
- 「sql」または「svchost」で始まるプロセスすべてに一致します。
^sql|^svchost

サービス自動ウォッチャー

インスタンス条件正規表現、正規表現 2、正規表現 3 :

- 表示名に「SQL」が含まれるすべてのサービスに一致します。
.*SQL.*
- 表示名が「SQL」で始まるすべてのサービスに一致します。
^SQL
- 大文字小文字を区別せずに「SQL」で始まるすべてのサービスに一致します。
/^SQL/i

関連項目：

[Perl 互換正規表現（PCRE）のサポート](#) (P. 178)

[複数行の正規表現](#) (P. 180)

[テキストパターン除外の設定](#) (P. 181)

[正規表現の例](#) (P. 181)

Perl 互換正規表現(PCRE)のサポート

Perl 互換正規表現 (PCRE) を使用すると、正規表現をサポートするモニタを定義するときに **i18n (UTF-8)** 互換の正規表現を指定できます。これらのモニタは、ログファイル、プロセス、プロセス グループ、**Windows** サービス、および **Windows** イベントをモニタします。また、このオプションを使用して、より複雑な正規表現を作成できます。

以下の 2 つの正規表現ライブラリのどちらかを選択できます。

- **SystemEDGE** の標準正規表現ライブラリ (デフォルト)
- **Perl 互換正規表現 (PCRE)** ライブラリ

PCRE は、区切り文字が異なり、特殊文字の範囲も広いため、標準正規表現ライブラリと **100 パーセント** の互換性はありません。

PCRE ライブラリを有効にするには、**SystemEDGE** のレガシー モードで手動で設定ファイルを編集し、以下の行を追加します。

```
use_pcre
```

設定ファイルを管理モードで編集すると、ポリシーを次に展開したときに設定が消去されます。

管理モードで **PCRE** ライブラリを有効にするには、**CA Virtual Assurance** の [ポリシー設定] ユーザ インターフェースを使用します。このユーザ インターフェースで、[制御設定] タブの [Perl 互換正規表現を使用] オプションを使用します。「<JS>」パターン修飾子を指定すると、この式の **JavaScript** 互換モードがアクティブになります。

PCRE は、**POSIX** 基本正規表現、**POSIX** 拡張正規表現、および Perl によって提供される正規表現機能のサブセットをカバーします。PCRE の詳細については、<http://www.pcre.org> を参照してください。

入力の正規表現としてサポートされる形式は以下のとおりです。

/regular expression/switch

regular expression

注: PCRE の使用方法としては、区切り文字とオプションのスイッチを含めて正規表現を指定する方法 (*/regex/i*) をお勧めします。

サポートされるスイッチのリスト

/i

PCRE_CASELESS (Perl 互換の */i*) : 大文字と小文字の両方に一致します。

/m

PCRE_MULTILINE (Perl 互換の */m*) : 複数行にわたって一致します。

/s

PCRE_DOTALL (Perl 互換の */s*) : パターン内のドット メタ文字が任意の値の文字 (改行を示す文字を含む) に一致します。

/x

PCRE_EXTENDED (Perl 互換の */x*) : パターン内の空白データ文字が、エスケープされた場合と文字クラスの内部にある場合を除いて無視されます。

/E

PCRE_DOLLAR_ENDONLY : \$ が末尾にのみ一致します。

/U

PCRE_UNGREEDY : 繰り返し限定記号の最長一致を反転します。つまり、デフォルトでは最長一致を行いませんが、後ろに疑問符を付けると最長一致を行います。

/<JS>

PCRE_JAVASCRIPT_COMPAT : JavaScript 互換を指定します。

サポートされる区切り文字のリスト

SystemEDGE では、デフォルトの正規表現区切り文字 (/) に加えて、UNIX および Linux のパスに含まれるスラッシュ記号を指定する式を簡略化するため、代替の区切り文字がサポートされています。代替の区切り文字は (#)、(|)、および (:) です。

たとえば、以下の正規表現は同等です。

```
/¥/dir1¥/dir2¥/dir3¥/filename/i  
#/dir1/dir2/dir/filename3#i  
|/dir1/dir2/dir3/filename|i  
:./dir1/dir2/dir3/filename:i
```

複数行の正規表現

Perl 互換正規表現 (PCRE) には、複数の行に表示されるテキストを照合する機能があります。この機能を使用するには、式の中で PCRE_MULTILINE (/m) または PCRE_DOTALL (/s) スイッチを使用します。

ログ ファイル モニタの正規表現でどちらかのスイッチが検出されると、エージェントはモニタされているログ ファイルの最後の 8 KB の範囲内で指定された式を検索します。

テキスト パターン除外の設定

Perl 互換正規表現 (PCRE) を使用して、正規表現内のテキスト パターン除外を設定できます。PCRE ライブラリでは、正規表現パターン マッチングの実行が有効になっています。PCRE ライブラリは、パターンマッチング時に包含または除外する正規表現内のパターンを指定できる柔軟性を備えています。

検索条件によっては、(ログ ファイル内の) 特定のテキスト パターンが除外されている場合にのみテキスト パターンに一致する必要があります。テキスト パターンを除外するには、一意の正規表現を作成し、除外されるパターンを含む式を指定します。たとえば、「def」を除外し、「abc」を含むサブ文字列との一致のみを識別する式を作成できます。正規表現で「def」を除外するには、件名にサブ文字列「def」が含まれる場合は一致しないことを示す否定の表明を使用します。

例: テキスト パターン除外の設定

以下の例は、文字列「def」が後続しない文字列「abc_」のすべてに一致するテキスト パターンを検索するための検索オプションを示しています。ここでは、かっこ () を使用して文字列をグループ化し、?! は 否定を表します。

```
abc_(?!def)
```

正規表現の例

このセクションでは、sysedge.cf ファイルの watch logfile ディレクティブを使用してログ モニタ テーブルに正規表現を追加する例を示します。

簡単な式

例: /the quick brown fox/

以下の例では、文章から正規表現「/the quick brown fox/」を検索します。

入力

```
The quick brown fox jumps over the lazy dog.
```

```
the quick brown fox jumps over the lazy dog.
```

結果

```
the quick brown fox
```

大文字と小文字を区別しない式

例: `/The quick brown fox/i`

以下の例では、文章から正規表現「`/The quick brown fox/i`」を検索します。この検索は、大文字と小文字を区別せずに式「`The quick brown fox`」に一致します。

入力

The quick brown fox jumps over the lazy dog.

the quick brown fox jumps over the lazy dog.

the quick brown FOX jumps over the lazy dog.

結果

The quick brown fox jumps over the lazy dog.

the quick brown fox jumps over the lazy dog.

the quick brown FOX jumps over the lazy dog.

複数行モード - 行頭および行末のメタ文字

例: `/^abc$/m`

以下の例では、文章から正規表現「`/^abc$/m`」を検索します。この式は、パターン「`abc`」が同じ行にある場合にのみ一致します。

入力

Lazy dog¥nquick horse¥nabc

結果

abc

複数行モード - PCRE_DOTALL

例: `/two.*three.*four/s`

以下の例では、文章から正規表現「`/two.*three.*four/s`」を検索します。
PCRE_DOTALL パターン修飾子を指定すると、ドットメタ文字が改行にも一致します。

入力

```
one number¥ntwo numbers¥nthree numbers¥nfour numbers¥nfive numbers
```

結果

```
Two numbers¥nthree numbers¥nfour
```

拡張された式 - 空白の無視

例: `/^1234 # comment in input string/x`

以下の例では、文章から正規表現「`/^1234 #comment in extended re/x`」を検索します。この式は、入力の中に空白があるかどうかに関係なく引用されたテキストに一致します。

入力

```
1234 #comment in input string
```

```
1234 #  comment in      input  string
```

```
1234#commentininputstring
```

結果

```
1234 #comment in input string
```

```
1234 #  comment in      input  string
```

```
1234#commentininputstring
```

ドル記号が末尾にのみ一致

例: `/X$/E`

以下の例では、文章から正規表現「`/X$/E`」を検索します。パターン内のテキスト（「`X`」）は、件名文字列の末尾にある場合にのみ一致します。

入力

Test X

What can I do with X

結果

X

X

JavaScript 互換

例: `/a[^]b/<JS>`

以下の例では、文章から正規表現「`/a[^]b/<JS>`」を検索します。「`<JS>`」パターン修飾子を指定すると、この式の JavaScript 互換モードがアクティブになります。

入力

aXb

結果

aXb

SNMPv3 設定

SNMPv3 は、以前のバージョンよりセキュリティのレベルが向上しています。SNMPv3 には以下の通信レベルがあります。

- **noAuthNoPriv** は、SNMPv1 および SNMPv2 とほぼ同じです。メッセージにユーザ名が添付され、送信者と受信者の間でユーザ名が一貫している必要があります。
- **AuthNoPriv** は、一貫したユーザ名とパスワードを使用します。
- **AuthPriv** は、ユーザ名、パスワード、および暗号化キーを使用します。このキーでメッセージの本体が暗号化されます。

情報の一貫性を維持するには、送信者と受信者の両方が同期し、用意されている暗号化ツールを使用して情報のセキュリティを確保する必要があります。

SNMPv3 ベースの通信を使用するように **SystemEDGE** を設定できます。エージェントは、SNMPv3 用に定義されたユーザベースセキュリティモデル (USM) に基づいています。

sysedgeV3.cf ファイルには、SNMP 管理者がどのように責務に対応し、ホストへのアクセス時に期待されるセキュリティのレベルを指定するかを定義したポリシーが含まれます。このファイルは **CASYSEDGE_DATADIR** ディレクトリにあり、以下を指定できます。

- SNMP エンジン ID プレフィックス
- SNMPv3 セキュリティ ユーザ
- セキュリティのレベル
- 認証プロトコルとパスワード
- 暗号化プロトコルとそのパスワード

sysedgeV3.cf ファイルを設定するためのユーザインターフェースはありません。このファイルに SNMPv3 の設定項目を直接設定する必要があります。ただし、CA Virtual Assurance マネージャからこのファイルにアクセスし、設定したファイルをシステムに展開することができます。

SNMPv3 エンジン ID の設定

テキスト形式の **SNMP エンジン ID** プレフィックスを指定する必要があります。このプレフィックスがエージェントの **SNMP** ライブラリによってプロセス ID および IP アドレスと連結されることにより、エージェントが **SNMPv3** で動作します。

SNMPv3 エンジン ID を設定するには、`sysedgeV3.cf` ファイルに以下の行を追加します。

```
SNMP_V3_ENGINE_ID <engineIdPrefix>
```

`engineIdPrefix` のデフォルト値は `SystemEDGEAdmin` です。設定する文字列にはスペースを使用しないでください。

たとえば、**SNMPv3** 設定ファイルの以下の行は、文字列 `CompanySNMPV3ADMIN` を **SNMP** エンジン ID プレフィックスとして指定します。

```
SNMP_V3_ENGINE_ID CompanySNMPV3ADMIN
```

SNMPv3 ユーザ情報の設定

`SNMP_V3_USER_INFO` キーワードを使用して `sysedgeV3.cf` ファイルに **SNMPv3** **USM** のユーザおよびセキュリティ情報を指定します。このキーワードに対するすべての引数は、1つの行に所定の順序で、スペースで区切って入力する必要があります。

SNMPv3 のユーザ情報を設定するには、`sysedgeV3.cf` ファイルに以下の行を追加します。

```
SNMP_V3_USER_INFO *|access[addresses] userName securityModel securityLevel  
[authProtocol authPassword [privProtocol privPassword]]
```

access

読み取りまたは書き込みアクセスを指定します。値として「**read**」または「**write**」のいずれかを指定する必要があります。

addresses

指定された IP アドレスまたはサブネットから送信された要求をフィルタする IP フィルタを指定します。このフィールドは必須ではありません。指定しなかった場合は、すべてのホストがエージェントの情報にアクセスできるようになります。詳細については、「**SNMPv3** ユーザのアドレス フィルタリング」を参照してください。

userName

アクセスを許可する **SNMPv3** セキュア ユーザの名前を指定します。

securityModel

使用する **SNMPv3** セキュリティ モデルを指定します。現時点では、**SystemEDGE** エージェントはユーザ ベース セキュリティ モデル (**USM**) のみをサポートします。値として **3** のみがサポートされています。

securityLevel

サポートされるセキュリティ レベルとして、以下の値がサポートされています。

noAuthNoPriv

この **SNMPv3** ユーザが使用する認証プロトコルとプライバシー (暗号化) プロトコルが設定されていないことを示します。

AuthNoPriv

この **SNMPv3** ユーザ用の認証プロトコルが設定され、プライバシー プロトコルが設定されていないことを示します。

AuthPriv

この **SNMPv3** ユーザが使用する認証プロトコルとプライバシー プロトコルが設定されていることを示します。

authProtocol

使用する認証プロトコルを指定します。現時点では、**MD5** プロトコルと **SHA** プロトコルのみが使用されます。このオプションは、セキュリティ レベルとして **AuthPriv** または **AuthNoPriv** を設定した場合にのみ指定する必要があります。

authPassword

認証プロトコルによって使用される **SNMPv3** ユーザの認証パスワード (キー) を指定します。**authPassword** は、**authProtocol** (**MD5** または **SHA**) を設定した場合にのみ指定する必要があります。

privProtocol

SNMPv3 ユーザが使用する暗号化 (プライバシー) プロトコルを指定します。サポートされているプロトコルは、**DES**、**3DES**、および **AES** のみです。暗号化プロトコルを指定する場合は、**authProtocol** と **authPassword** も指定する必要があります。**privProtocol** を指定した場合は、セキュリティ レベルとして **AuthPriv** のみがサポートされます。

privPassword

暗号化プロトコルによって使用される **SNMPv3** ユーザの暗号化パスワード（キー）を指定します。このパラメータは、**privProtocol** を設定した場合にのみ必要です。

各セキュリティ レベルに読み取りまたは書き込みアクセスを割り当てることができます。たとえば、認証なし、プライバシーなし（**noAuthNoPriv**）のセキュリティ レベルをパブリック コミュニティ文字列と同等にし、認証およびプライバシー（**AuthPriv**）を管理コミュニティ文字列と同等にすることができます。

例

有効な SNMPv3 ユーザ定義の例を以下に示します。

```
SNMP_V3_USER_INFO *|read joe1 3 AuthPriv MD5 apass AES ppass
SNMP_V3_USER_INFO *|read joe2 3 AuthPriv SHA apass DES ppass
SNMP_V3_USER_INFO *|write|192.168.29.0 joe3 3 AuthPriv SHA apass 3DES ppass
SNMP_V3_USER_INFO
*|write|100.10-255.100.101,e000-ffff:f0ff:bef0:*,192.168.120.0,*:1 joe4 3
AuthNoPriv SHA evansar
SNMP_V3_USER_INFO *|read joe5 3 noAuthNoPriv
```

SNMPv3 ユーザのアドレス フィルタリング

SNMPv3 設定ファイルでは、ソース アドレス フィルタを指定して、情報へのアクセスをアドレス フィルタに一致するソース アドレスのみに制限できます。

アドレス フィルタリング フィールドに関してサポートされる構文と使用ガイドラインを以下に示します。

- 完全な IPv4 アドレスがサポートされています。

例：

`130.10.100.101`

この行は、ソース アドレス `130.10.100.101` からの要求に対してエージェントへのアクセスを許可することを指定します。

- 完全な IPv6 アドレスがサポートされています。IPv6 アドレスの 16 進数表記を指定します。文字は小文字または大文字で指定できます。IPv6 アドレスの例を以下に示します。

`Ea2f:fe90:abcd:0000:230:a2f:200:ad01`

この行は、ソース アドレス `Ea2f:fe90:abcd:0000:230:a2f:200:ad01` からの要求に対してエージェントへのアクセスを許可することを指定します。

- ホスト名がサポートされています。ただし、サーバに複数の IP アドレスが関連付けられている場合があり、DNS ですべての IP アドレスがそのホスト名に関連付けられているとは限らないため、ホスト名ではなく IP アドレスをフィルタとして使用することを強くお勧めします。

例：

`box1.domain.com`

この行は、`box1.domain.com` からの要求に対してエージェントへのアクセスを許可することを指定します。

注：ホスト名を指定する場合は、エージェントが受信した IP アドレスを指定されたソース ホスト名に解決できる必要があります。

- ワイルドカード文字としてアスタリスク (*) がサポートされています。ワイルドカード文字を使用して、エージェントにアクセスできる一連のアドレスを指定します。

例：

`130.10.*.101`

この行は、`130.10` で始まり `101` で終わるソース IPv4 アドレス（つまり、`130.10.0.101` ～ `130.10.255.101`）からの要求に対してエージェントへのアクセスを許可することを指定します。

例：

Ea2f:fe90*:ad01

この行は、ea2f:fe90 で始まり ad01 で終わるソース IPv6 アドレスからの要求に対してエージェントへのアクセスを許可することを指定します。

- ワイルドカード文字 (*) を指定する場合は、その IP フィールド内で唯一の文字である必要があります。IP フィールドとは、ドット「.」（IPv4 アドレスの場合）およびコロン「:」（IPv6 アドレスの場合）の間にある文字のことです。

たとえば、130.10.3*.200 というエントリは、3 とアスタリスクを同じドット間隔の中に置くことができないため、無効です。

Ea2f:fe*:abcd:0000:230:a2f:200:ad01 というエントリも、「fe」とアスタリスクを同じコロン間隔の中に置くことができないため、無効です。

- ワイルドカードを複数個使用した場合は、欠落するフィールドを補充するために左端のワイルドカードのみが拡張されます。たとえば、2002*:12f4*:1012 と入力するのは 2002*:*:12f4*:1012 と入力するのと同じことです。
- 0（ゼロ）はワイルドカード文字として扱われます。

例：

130.10.120.0

この行は、IPv4 サブネット 130.10.120 からの要求に対してエージェントへのアクセスを許可することを指定します。このようにアドレスを入力するのは 130.10.120 を指定するのと同じことです。

例：

Ea2f:fe90:0:ad01

この行は、ea2f:fe90 で始まり ad01 で終わる IPv6 アドレスからの要求に対してエージェントへのアクセスを許可することを指定します。これは Ea2f:fe90*:ad01 と入力するのと同じことです。

- 1つのゼロを持つワイルドカードの性質を取り除き、ゼロをリテラルとして扱うには、ゼロを複数個使用します。

例：

130.000.120.00

この行は、IPv4 アドレス 130.0.120.0 からの要求に対してエージェントへのアクセスを許可することを指定します。

例：

Ea2f*:0000:000:ad01

この行は、ea2f で始まり 0:0:ad01 で終わる IPv6 アドレスからの要求に対してエージェントへのアクセスを許可することを指定します。

- 部分的な IP アドレスがサポートされています。部分的なアドレスは、ワイルドカード (*) が後に続く部分的なアドレスとして扱われます。その部分的なアドレスで始まるソース アドレスから送信されたすべての要求が許可されます。

たとえば、130.10 は 130.10.* と指定するのと同じです。また、Ea2f:fe90 は Ea2f:fe90:* と指定するのと同じです。

- IPv6 アドレスの場合、連続する 2 つのコロン (::) は 2 つのコロンに挟まれたワイルドカード (:*) として扱われます。
- ダッシュ (-) で表すアドレス範囲は、IPv4 アドレスと IPv6 アドレスでサポートされています。任意またはすべてのフィールドで範囲を定義できます。

例：

130.10-255.100.101

この行は、130.10 ～ 130.255 のソース IPv4 アドレスからの要求に対してエージェントへのアクセスを許可することを指定します。

e000-ffff:f0ff:bef0:*

この行は、e000:* ～ efff:* のソース IPv6 アドレスからの要求に対してエージェントへのアクセスを許可することを指定します。

- カンマで区切った複数のフィルタがサポートされています。

例：

130.10-255.100.101, e000-ffff:f0ff:bef0:*, 130.10.120.0, box2.domain.com

- スペースは使用できません。

注：ホストに複数のインターフェースまたは IP アドレスがあるときは、要求 (snmpget、walktree、snmpset など) をどのインターフェースまたは IP アドレスからも送信できるため、エージェントにアクセスするためにすべてのインターフェースおよび IP アドレスを指定する必要があります。ホストに割り当てられたすべての IP アドレスをエージェントの IP フィルタに定義する必要があります。そうしないと、ホストからエージェントにアクセスできません。すべての IP アドレスとインターフェースを定義する代わりに、ホスト名を指定することもできますが、ホスト名を使用する場合の制限事項に注意する必要があります。

SNMPv2c および SNMPv3 のトラップ デスティネーションの設定

`sysedgeV3.cf` ファイルでは、エージェントが送信するトラップのタイプとトラップのプロパティを指定できます。サポートされているトラップのタイプは、以下のとおりです。

- SNMPv2c トラップ
- SNMPv2c 通知（INFORM 要求および確認済みトラップとも呼ばれる）
- SNMPv3 トラップ
- SNMPv3 通知（INFORM 要求および確認済みトラップとも呼ばれる）

注: SNMPv1 トラップ デスティネーションは、`sysedgeV3.cf` ではなく `sysedge.cf` で設定します。

SNMPv2c トラップおよび SNMPv2c 通知は、SNMP のコミュニティ文字列を使用して送信されます。このコミュニティ文字列は、`sysedge.cf` で定義する必要があります（たとえば、`community global read-only`）。

SNMPv3 トラップおよび SNMPv3 通知は、SNMPv3 ユーザの認証情報を使用して送信されます。SNMPv3 ユーザは、`sysedgeV3.cf` 設定ファイルのトラップ定義レコードの前に定義する必要があります。

SNMPv2 トラップを指定されたデスティネーション ホストに送信するようにエージェントを設定するには、`sysedgeV3.cf` ファイルに以下の行を追加します。

```
SNMP_V2_TRAP_INFO <destination_host>|<port> <trap_context> <community>  
<trap-encoding>
```

SNMPv2c 通知トラップ（確認済みトラップおよび INFORM 要求とも呼ばれる）を指定されたデスティネーション ホストに送信するようにエージェントを設定するには、`sysedgeV3.cf` ファイルに以下の行を追加します。

```
SNMP_V2_NOTIFICATION_INFO <destination_host>|<port> <trap_context> <community>  
<timeout> <num_of_retries> <trap-encoding>
```

SNMPv3 トラップを指定されたデスティネーション ホストに送信するようにエージェントを設定するには、`sysedgeV3.cf` ファイルに以下の行を追加します。

```
SNMP_V3_TRAP_INFO <destination_host>|<port> <trap_context> <SNMPv3_user>  
<trap-encoding>
```


SNMPv3 通知トラップ（確認済みトラップおよび **INFORM** 要求とも呼ばれる）を指定されたデスティネーションホストに送信するようにエージェントを設定するには、`sysedgeV3.cf` ファイルに以下の行を追加します。

```
SNMP_V3_NOTIFICATION_INFO <destination_host>|<port> <trap_context> <SNMPv3_user>  
<timeout> <num_of_retries> <trap-encoding>
```

destination_host

トラップの送信先となるホストを指定します。ホスト名または IP アドレスを指定できます。

port

トラップの送信先となるデスティネーションホスト上のポート番号を指定します。

trap_context

このフィールドの値としては、*（アスタリスク）のみがサポートされています。この値は必須です。

community

（**SNMPv2c** のみ）トラップの送信時に使用する **SNMP** コミュニティ文字列を指定します。この文字列は、`sysedge.cf` ファイルでも定義する必要があります。

SNMPv3_user

（**SNMPv3** のみ）トラップの送信時に使用する **SNMPv3** ユーザを指定します。このユーザは、`sysedgeV3.cf` ファイルでも定義する必要があります。

timeout

（オプション、通知のみ）通知の配信確認処理がタイムアウトするまでの待機時間を秒単位で指定します。

num_of_retries

（オプション、通知のみ）タイムアウト後に通知の送信を再試行する回数を指定します。

trap-encoding

（オプション）トラップの送信時に使用するエンコーディングのタイプを指定します。これは、**SNMPv1** のトラップエンコーディングの設定とほぼ同じです。詳細については、[「SNMPv1 トラップデスティネーションの設定 \(P. 152\)」](#)を参照してください。

例

以下の例では、SNMPv3 ユーザ `osmanager` を対象としてホスト `sysmanager` 上のポート `6666` に SNMPv3 トラップ通知を送信し、タイムアウトを `30` 秒に設定して確認応答を待機し、最大 `2` 回再試行します。

```
SNMPV3_NOTIFICATION_INFO sysmanager|6666 * osmanager 30 2
```

以下の例では、プライベート コミュニティを使用してホスト `localhost` 上のポート `162` に SNMPv2c トラップを送信します。

```
SNMPV2_TRAP_INFO localhost|162 * private
```

以下の例では、パブリック コミュニティを使用してホスト `localhost` 上のポート `162` に SNMPv2c トラップ通知を送信し、タイムアウトを `30` 秒に設定して確認応答を待機し、最大 `3` 回再試行します。

```
SNMPV2_NOTIFICATION_INFO localhost|162 * public 30 3
```

SNMPv3 設定ファイルの暗号化

SNMPv3 の暗号化は、CA が定義したデフォルトのキーおよび、暗号化（プライバシー）プロトコルとしては DES を、認証プロトコルとしては SHA を使用して提供されます。暗号化された設定ファイルを一元化された場所に生成し、複数のホストで共有できます。

セキュリティを強化するため、SNMPv3 設定ファイル `sysedgeV3.cf` を暗号化できます。このファイルのクリアテキストバージョンを作成して暗号化し、暗号化されたファイルを適切なディレクトリにインストールする必要があります。

オプションの `-L` スイッチを使用すると、コンソールの現在のロケールと言語カタログ（利用可能な場合）が検出されます。言語カタログが見つからない場合は、デフォルト言語の英語が使用されます。

SNMPv3 設定ファイルを暗号化する方法

1. `sysedgeV3.cf` の構文をテストし、検証します。
2. 以下のコマンドを使用して `sysedgeV3.cf` を暗号化します。

```
se_enc -i sysedgeV3.cf -o sysedgeV3.cf.crypt -m 2 -d 2
```
3. クリアテキストの `sysedgeV3.cf` をアーカイブ領域に移動します。
4. `sysedgeV3.cf.crypt` の名前を `sysedgeV3.cf` に変更します。
5. `sysedgeV3.cf` をエージェントのデータ ディレクトリにコピーするか、または CA Virtual Assurance マネージャからのリモート展開でこのファイルを適用します。

注: SNMPv3 設定ファイルは複数のホスト間で共有できます。

SNMPv1 および SNMPv2c の無効化

エージェントの SNMPv1 および SNMPv2c 通信を無効にするには、`sysedge.cf` ファイルで以下を含む行をコメントアウトまたは削除します。

```
community <name> <access> <addresses>
```

Windows の SystemEDGE コントロール パネル

SystemEDGE には、専用の SystemEDGE コントロール パネルがスタンドアロンの Windows サービスとして用意されています。SystemEDGE コントロール パネルを表示するには、[コントロール パネル] ダイアログ ボックスを開き、[SystemEDGE] をダブルクリックします。

SystemEDGE コントロール パネルを使用して以下のタスクを実行できます。

- エージェントの起動と停止
- コミュニティ文字列およびトラップ デスティネーションの表示
- 設定ファイルのオープン
- sysedge.log ファイルの表示
- diagsysedge ユーティリティの実行
- 「ユーザ ガイド」と「リリース ノート」の表示

第 6 章: エージェントの起動

この章では、SystemEDGE エージェントの起動方法について説明します。その前に、エージェントを環境に合わせて設定する必要があります。詳細については、「エージェントの設定」の章を参照してください。

注: この章の説明に従ってエージェントを起動した後は、diagsysedge.exe プログラムを使用してエージェントが実行されていることを確認できます。詳細については、「トラブルシューティングと使用方法に関する推奨事項」の章を参照してください。

このセクションには、以下のトピックが含まれています。

[Windows システムでのエージェントの起動または停止](#) (P. 197)

[UNIX システムでのエージェントの起動または停止](#) (P. 198)

[コマンドライン起動オプション](#) (P. 200)

[システム ブート時のエージェントの自動起動](#) (P. 201)

[エージェントのウォーム スタート](#) (P. 202)

[保守モード](#) (P. 203)

Windows システムでのエージェントの起動または停止

Windows システムでは、コマンドライン、[サービス] コントロールパネル、または SystemEDGE コントロールパネルから手動で SystemEDGE を起動できます。

コマンドラインからエージェントを起動または停止するには、以下のリストから適切なコマンドを選んで入力します。

```
net start sysedge  
net stop sysedge
```

エージェントは、バックグラウンドでサービスとして起動し、インストール時に指定されたポートとエージェントのデータ ディレクトリに含まれるデフォルトの設定ファイルを使用します。

デフォルト以外のオプションやデフォルト以外の設定ファイルを使用してフォアグラウンドでエージェントを実行することもできます。詳細については、「[コマンドライン起動オプション](#) (P. 200)」を参照してください。

[サービス]コントロール パネルからエージェントを起動(停止)する方法

1. [スタート] - [管理ツール] - [サービス] を選択します。
Windows の [サービス] ダイアログ ボックスが表示されます。
2. SystemEDGE を右クリックして [開始] ([停止]) を選択します。
エージェントが起動 (停止) します。

SystemEDGE コントロール パネルからエージェントを起動する方法

1. コントロール パネルにアクセスし、[SystemEDGE] をダブルクリックします。
[SystemEDGE コントロール パネル] が表示されます。
2. [エージェントの起動] または [エージェントの停止] をクリックします。
エージェントが起動 (停止) します。

注: Windows XP のカテゴリ表示では、[SystemEDGE コントロール パネル] は [ネットワークとインターネット接続] の下にあります。Windows XP のクラシック表示では、[SystemEDGE コントロール パネル] アイコンは表示のメイン レベルにあります。

UNIX システムでのエージェントの起動または停止

UNIX システムでは、付属のサービス起動スクリプトを使用して SystemEDGE を起動または停止する必要があります。ごくまれに (CA サポートから指示された場合など)、エージェントを手動で直接開始することもあります。

UNIX システム用のサービス起動スクリプト

UNIX システムでは、サービス起動スクリプトを使用してエージェントを自動的に開始、停止、または再起動できます。エージェントはインストール時に指定したポート番号で起動されます。

すべての UNIX プラットフォーム用のサービス スクリプトを以下に示します。

`CASYSEDGE/bin/sysedgectl [start|stop|status|restart|probe]`

UNIX 上のコマンドラインからエージェントを開始するには、`Install_Dir/bin` ディレクトリから以下のコマンドを実行します。

`./sysedgectl start`

エージェントは、インストール時に指定されたポートとエージェントのデータディレクトリに含まれるデフォルトの設定ファイルを使用して起動します。

デフォルト以外のオプションやデフォルト以外の設定ファイルを使用してバックグラウンドでエージェントを実行することもできます。詳細については、「[コマンドライン起動オプション](#) (P. 200)」を参照してください。

例

```
cd $CASYSEDGE/bin  
  
./sysedgectl start  
./sysedgectl restart  
./sysedgectl stop
```

コマンドライン起動オプション

このセクションでは、Windows および UNIX システムで SystemEDGE エージェントを起動するためのコマンドライン オプションについて説明します。これらのオプションにより、デフォルト以外のポートや設定ファイルの値を使用してエージェントを開始できます。これらのオプションを使用して、コマンドシェルからエージェントの別のインスタンス（設定変更のテストなど）を実行できます。

注: エージェントを起動すると、既存のログ ファイルは「*filename.0*」のようにゼロ拡張子を付けて保存されます。管理モードで新しい設定を展開することによって実行時に既存のエージェント ログ ファイルが変更された場合、そのファイルはゼロ拡張子を付けて保存されます。

重要: エージェントのインストールと CA Virtual Assurance の展開では、デフォルトのエージェント インスタンスのみがサポートされます。

`sysedge` バイナリは以下コマンドライン オプションをサポートします。

`[-p port] [-f config file] [-e SNMPV3 config file] [-m monitor file] [-h] [-b] [-d]`

-p port

受信 **SNMP** メッセージをリスンするポートを指定します。エージェントの別のインスタンスを実行するには、インストール時に入力したポートとは異なるポートを指定します。**SystemEDGE** エージェントを実行するために使用できる代替ポートとして、ポート **UDP/1691** が予約されています。インストール時に指定したポートを使用する場合は、この引数を指定しないでください。

デフォルトのエージェント インスタンスのみが **CA Virtual Assurance** マネージャと通信します。したがって、**CA Virtual Assurance** から受信した設定ファイルは、デフォルトのポートに対応する **CASYSEDGE_DATADIR** ディレクトリに配置されます。

-f config file

CASYSEDGE_DATADIR 内のデフォルトの設定ファイルの代わりに、**sysedge.cf** ファイルに使用するパス名を指定します。

-e SNMPV3 config file

CASYSEDGE_DATADIR 内のデフォルトの **SNMPv3** 設定ファイルの代わりに、**sysedgeV3.cf** ファイルに使用するパス名を指定します。

-m monitor file

CASYSEDGE_DATADIR 内のデフォルトのモニタ テーブル設定ファイルの代わりに、**sysedge.mon** ファイルに使用するパス名を指定します。

-h

コマンドのヘルプを表示します。このオプションにより、利用可能なコマンドラインオプションが一覧表示されます。

-b

(UNIX のみ) エージェントをバックグラウンドで実行します。エージェントはデーモンプロセスとして実行され、制御端末から切断されます。このフラグは、起動スクリプトからエージェントを起動するときに使用します。

-d

エージェントをデバッグモードで実行します。このオプションにより、エージェントのロギングレベルが **debug3** に設定されます。ロギングメカニズムおよびロギングレベルの詳細については、「エージェントの設定」の章を参照してください。

システム ブート時のエージェントの自動起動

SystemEDGE は、システムのブート時に常にスクリプトを使用してエージェントを自動起動できるように、スクリプトをシステム起動領域に自動的にコピーします。

システムによっては、システムのブート時に常にエージェントを自動起動するために、追加の手順が必要な場合があります。

このセクションには、以下のオペレーティング システム用の手順が含まれます。

- Windows
- UNIX

注: エージェントを起動すると、既存のログ ファイルは「*filename.0*」のようにゼロ拡張子を付けて保存されます。管理モードで新しい設定を展開することによって実行時に既存のエージェントログ ファイルが変更された場合、そのファイルはゼロ拡張子を付けて保存されます。

Windows システムでのエージェントの自動起動

SystemEDGE サービスは、デフォルトでシステムのブート時に起動します。

エージェントが自動起動するように設定されていることを確認する方法

1. Windows の [サービス] ダイアログ ボックスを開きます。
2. [CA SystemEDGE] を右クリックし、[プロパティ] を選択します。
3. [全般] タブの [スタートアップの種類] ドロップダウン リストで [自動] が選択されていることを確認します。

UNIX システムでのエージェントの自動起動

UNIX システム上の SystemEDGE エージェントは、デフォルトでシステムのブート時にサービス起動スクリプトを使用して起動します。インストール時に指定したデフォルト ポートに接続し、インストールパスを認識します。

エージェントのウォーム スタート

SystemEDGE エージェントは、CA Virtual Assurance と統合することで、展開された設定ファイルを実装するためにコールドリスタートを実行する必要がなくなります。代わりに、CA Virtual Assurance から設定変更を受信することでウォーム スタートがトリガされ、エージェントは実行状態のまま変更を実装できます。エージェントは、ウォーム スタートの実行時にインターフェースの破棄と再バインド、OID 変数の再設定、および AIM の再起動を行う必要がないため、リソース消費の少ないすばやい設定変更が可能です。

CA Virtual Assurance を使用してランタイム設定変更を展開する方法の詳細については、「CA Virtual Assurance 管理ガイド」を参照してください。

ウォーム スタートの例外

SystemEDGE エージェントは、CA Virtual Assurance と統合することで、展開された設定変更の大部分を受信したときにウォーム スタートを実行します。ただし、以下の状況ではエージェントは引き続き完全にリサイクルされます。

- ユーザ コマンドを使用して手動でリサイクルした場合。コマンドラインまたは Windows 上の SystemEDGE コントロール パネルを使用して、いつでもエージェントを起動および停止できます。
- 設定ファイルの変更を受信し、エージェントの再起動が必要になったときに暗黙的にリサイクルされる場合。AIM の削除や `sysedge_fips_mode` パラメータの変更によって、コールドリスタートがトリガされます。エージェントは、これらの変更によってトリガされたプロセスを使用して自動的に再起動します。

注: CAM をアップグレードした場合も、エージェントの自動再起動がトリガされます。

- CAM を手動で停止したときに暗黙的にリサイクルされる場合 (SystemEDGE が CAM のクライアントとして登録されているため)。この場合、エージェントは自動的に再起動しません。CAM が再度実行された後、手動でエージェントを起動する必要があります。

保守モード

SystemEDGE エージェントには保守モードが用意されています。保守モードでは、エージェントはメトリックの収集と SNMP 要求への応答を継続しますが、モニタの処理や状態変更トラップの送信は行いません。このモードは、計画休止や保守を行うことがわかっており、その期間中にトラップを受信したくない場合に有用です。この場合、エージェントを保守モードに切り替えることで、エージェントの停止と再起動を回避できます。

保守モードを有効にする方法の詳細については、「エージェントの設定」の章を参照してください。

第 7 章: Systems Management Empire MIB

この章では、企業固有の Systems Management Empire MIB から入手可能な管理情報の例について説明します。

このセクションには、以下のトピックが含まれています。

[Systems Management Empire MIB の概要](#) (P. 206)

[システム グループ](#) (P. 207)

[ユーザ テーブル](#) (P. 209)

[グループ テーブル](#) (P. 209)

[プロセス テーブル](#) (P. 209)

[Who テーブル](#) (P. 210)

[リモート シェル グループ](#) (P. 211)

[パフォーマンス グループ](#) (P. 212)

[プロセス間通信グループ](#) (P. 213)

[バッファ グループ](#) (P. 214)

[ディレクトリ名のルックアップ キャッシュ グループ](#) (P. 215)

[AIX 論理パーティショングループ](#) (P. 216)

[ワーク ロード マネージャ \(WLM\) クラスのテーブル](#) (P. 216)

[トラップ コミュニティ テーブル](#) (P. 216)

[NT システム グループ](#) (P. 217)

[分散システム グループ](#) (P. 221)

[履歴テーブル](#) (P. 223)

[ログ モニタ テーブル](#) (P. 224)

[ディスク統計テーブル](#) (P. 224)

[CPU 統計テーブル](#) (P. 226)

[拡張グループ](#) (P. 227)

[プロセス モニタ テーブル](#) (P. 227)

[\[プロセス グループ モニタ\] テーブル](#) (P. 228)

[集計状態テーブル](#) (P. 228)

[LPAR グループ](#) (P. 228)

[ミラー テーブル](#) (P. 229)

[自動ウォッチャー テーブル](#) (P. 229)

[サポートされていない Systems Management Empire MIB オブジェクト](#) (P. 229)

Systems Management Empire MIB の概要

Systems Management Empire MIB モジュールでは、ホスト システムを管理するオブジェクトのコレクションを定義します。MIB はグループおよびテーブル別に整理され、以下のタイプの情報をモニタします。

- システム情報
- マウントされたデバイス
- カーネル設定
- ストリーム
- ユーザおよびグループ
- プロセス
- リモート コマンドの実行
- パフォーマンス
- プロセス間通信 (IPC)
- ネットワーク バッファのバッファ統計
- ストリーム バッファ
- I/O バッファ キャッシュ
- ディレクトリ名のルックアップ キャッシュ
- 論理パーティション
- NFS 統計および RPC 統計
- Windows 固有の統計
- ディスク統計および CPU 統計

MIB には、モニタリング テーブルとオブジェクトの集計テーブルも用意されています。セルフ モニタリング エントリを使用する MIB グループおよびテーブルからの特定のオブジェクトをモニタできます。

注: ローカル セキュリティ ポリシーに基づいて、任意のテーブルのサポートを無効にできます。テーブルのサポートを無効にする方法の詳細については、「エージェントの設定」の章を参照してください。

オブジェクト OID またはオブジェクト クラス、インスタンス、および属性の指定により、セルフ モニタリング エントリを作成できます。セルフ モニタリング機能を使用して、定義されたしきい値に対して任意の MIB オブジェクトをモニタします。取得した現在値に基づいたオブジェクト状態を保持できます。しきい値に違反すると、エージェントは状態変更トラップを送信します。詳細については、「セルフ モニタリング」の章を参照してください。

Systems Management Empire MIB のオブジェクトのトップレベル OID は 1.3.6.1.4.1.546 です。完全な OID を取得するには、MIB ファイルを参照してください。

この章には、以下の情報が含まれます。

- Systems Management Empire MIB のすべてのグループおよびテーブルのリスト
- MIB グループおよびテーブルのデフォルト クラスの値
- MIB テーブルのデフォルト インスタンスの値

モニタリング エントリを定義するには、各グループおよびテーブルのデフォルト クラスとデフォルト インスタンスの値を使用します。syedge.oid ファイルは、OID を指定せずに、セルフ モニタ エントリを定義するために使用される、各属性のデフォルト値を保持します。

この章では、すべてのグループとテーブルのデフォルト クラス、およびすべてのテーブルのデフォルト インスタンスの定義について説明します。MIB グループの属性をモニタする場合、インスタンス値は不要です。デフォルトの属性値は MIB 属性の名前です。

注: OID の逆引きルックアップ機能の詳細については、「セルフ モニタリング」の章を参照してください。

MIB オブジェクトおよび各属性名の詳細については、エージェントのインストールディレクトリの mib サブディレクトリ内にある empire.asn1 ファイルを参照してください。

システム グループ

基本情報をモニタするには、sysedge システム グループ内のオブジェクト値を取得します。

クラス : sysedgeSystem

デバイス テーブル

デバイス テーブルは、ホスト システムにマウントされているデバイスおよびファイル システムに関する情報を提供します。

クラス : devTableEntry

インスタンス : 特定のデバイスの devMntPt 属性値

ファイル システム領域のモニタリング

デバイス テーブルの [devCapacity] 列には、ファイル システム領域の使用率が表示されます。ファイル システム領域の使用率がいっぱいかどうかを確認するには、[devCapacity] 列を使用します。

セルフ モニタ テーブルを使用して、エージェントに対し、devCapacity 値をモニタして、devCapacity 値がしきい値を超えた場合には、管理システムにトラップを送信するように指示します。たとえば、以下のディレクティブを使用して、/usr ディレクトリをモニタし、/usr ディレクトリの使用容量が 90% を超えた場合に「重大」の状態変更トラップを送信するようにエージェントを設定します。

```
monitor 11 0x00 60 absolute > 90 'Monitor device space usage' '' devTableEntry '/usr' devCapacity
```

しきい値の設定の詳細については、「セルフ モニタリング」の章を参照してください。

マウントされたデバイスのマウント解除

マウントされたデバイスをマウント解除するには、そのデバイスの devUnmount 列に対して **SNMP Set** コマンドを発行し、列の値を **delete (1)** に設定します。エージェントによってデバイスがマウント解除され、デバイス テーブルから該当するエントリが削除されます。

注: SNMP Set 操作の場合、読み取り/書き込みアクセス権を持つコミュニティ名を使用してください。

カーネル設定グループ

システム上で実行されているカーネルバージョンを識別し、カーネル設定の詳細を確認するには、カーネル設定グループ内のオブジェクトを取得します。

クラス : kernelConfig

ブート設定グループ

ルート ファイル システムに使用されているデバイスおよびパーティションを確認するには、ブート設定グループの **MIB** オブジェクトを使用します。また、ダンブ ファイル システム、スワップ領域、および各システムに含まれるブロックの数も確認できます。

クラス : bootconf

ストリーム グループ

ストリーム I/O サブシステムは、ホスト内でユーザ スペースのアプリケーションとカーネル スペースのドライバ間にデータ転送パスまたはデータ処理パスを提供します。ストリーム グループ内のオブジェクトを取得することによって、ストリーム サブシステムの健全性をモニタできます。

クラス : streams

ユーザ テーブル

システム上で作成されたユーザ アカウントに関する情報を取得するには、ユーザ テーブルを使用します。このテーブルの各行はユーザ アカウントを表します。

クラス : userEntry

インスタンス : 特定のユーザの userLoginID 属性値

グループ テーブル

ホスト システム上で作成されたユーザ グループに関する情報を取得するには、グループ テーブルを使用します。テーブル内の各行は、`/etc/group` ファイルで定義されたグループを表します。

クラス : groupEntry

インスタンス : 特定のグループの groupName 属性値

プロセス テーブル

システム上で現在実行されているプロセスおよびプロセス メトリックを確認するには、プロセス テーブルを使用します。

クラス : processEntry

インスタンス : 特定のプロセスの processName 属性値

プロセスの nice 値を変更する (UNIX のみ)

プロセスの **nice** 列の値を設定または変更するには、特定のプロセスの **processNice** 列に対して **SNMP Set** コマンドを発行します。たとえば、プロセスの **nice** 列に優先度の値を設定することにより、プロセスのスケジューリング優先度を増減できます。

値のリストについては、UNIX の **nice (1)** マニュアル ページ、Windows の場合は **MSDN Web** サイトで **SetPriorityClass ()** 関数の説明を参照してください。

プロセスにシグナルを送信する

プロセスにシグナルを送信するには、特定のプロセスの **processKill** 列に対して **SNMP Set** コマンドを発行し、適切なシグナル値を設定します。

注: Windows では、シグナル番号 **SIGKILL (9)** のみがサポートされています。

たとえば、対象となる **processKill** 列の値を **9** に設定することにより、許可されていないプロセスをリモート ネットワーク管理ステーション コンソールから終了できます。**9** は UNIX **SIGKILL** シグナルを表し、プロセスを強制終了できます。

シグナルの詳細については、UNIX の **signal (3V)** マニュアル ページを参照してください。

Who テーブル

Who テーブルを使用して、ホスト システムに現在ログオンしているユーザのリストを取得できます。このテーブルは、特定のシステムの特定の時間のユーザ詳細を取得するために使用されます。このテーブルの各行は現在のユーザを表します。

クラス : **whoEntry**

インスタンス : 特定のユーザとデバイスの **whoName** 属性値および **whoDevice** 属性値

リモート シェル グループ

リモート ホスト システム上でシェル スクリプトやプログラムを実行するには、リモート シェル グループを使用します。

クラス : remoteShell

リモート シェル グループの MIB 変数を使用して、以下のコマンドを指定できます。

- コマンド構文 (shellCmd)
- 出力ファイル (shellOutput)

リモート シェル グループの変数を使用して、以下のコマンド情報を表示できます。

- シェル コマンドの実行からの終了コード (shellExitstat)
- 出力ファイルからのコマンド出力 (shellOutputContents)

リモート コマンドの実行

リモート コマンドを実行するには、MIB 変数 shellCmd に対して SNMP Set を実行します。SNMP Set 操作を実行するときは、読み取り/書き込みアクセス権を持つコミュニティ名を使用してください。

リモート コマンドを実行するには、コマンドプロンプトで次のコマンドを入力します。

```
SET shellOutput.0 = 出力ファイル名
SET shellCmd.0 = コマンド
```

エージェントは指定されたコマンドの fork/execs を実行し、MIB 変数 shellOutput で指定されたファイルに標準出力 (stdout) および標準エラー (stderr) をリダイレクトします。コマンドの実行が完了すると、shellExitStat にコマンドの終了ステータスが格納されます。

パフォーマンス グループ

ホストのオペレーティング システムのヘルスおよびパフォーマンスを追跡するには、パフォーマンス グループを使用します。

クラス : `kernelperf`

一定の期間にわたってサンプリングされる統計をモニタするには、パフォーマンス グループ属性を使用します。

注: サンプリング間隔はシステムによって異なります。

上記の MIB オブジェクトの詳細については、エージェントの配布ディレクトリの `mib` サブディレクトリ内にある `empire.asn1` ファイルを参照してください。

プロセス間通信グループ

プロセス間通信 (IPC) グループから以下のいずれかのテーブルを取得することにより、システム上で使用されている IPC メカニズムを追跡できます。

メッセージ キュー テーブル

現在使用中のシステム メッセージ キューに関する情報を提供します。

クラス : **msgqueEntry**

インスタンス : 特定のメッセージ キューの **queID** 属性値

共有メモリ テーブル

現在使用中のシステム共有メモリ セグメントに関する情報を提供します。

クラス : **shmemEntry**

インスタンス : 特定の共有メモリ セグメントの **shmemID** 属性値

セマフォ テーブル

現在使用中のシステム セマフォに関する情報を提供します。セマフォとは、オペレーティング システムまたはカーネル ストレージが持つ値です。各プロセスでは、この値を確認および変更して、複数のプロセスが同じオペレーティング システム リソースをめぐって競合した場合に、アクティビティを調整できます。

クラス : **semEntry**

インスタンス : 特定のセマフォの **semID** 属性値

上記のテーブルの各行は、IPC の使用状況の各インスタンスを表します。列はテーブルごとに異なります。

IPC メカニズムの削除

メッセージキュー、共有メモリ セグメント、およびセマフォを削除するには、IPC グループ テーブルを使用します。各 IPC グループ テーブルには、IPC 使用量のインスタンスを破棄するためにエージェントが使用する列が含まれます。

IPC メカニズムを削除するには、`queDel` 属性（メッセージキュー テーブル）、`shmemDel` 属性（共有メモリ テーブル）、または `semDel` 属性（セマフォ テーブル）に対して **SNMP Set** コマンドを発行し、属性値を **delete (1)** に設定します。

注：SNMP Set 操作の場合、読み取り/書き込みアクセス権を持つコミュニティ名を使用してください。

バッファグループ

バッファグループには、以下のバッファタイプをモニタするためのグループおよびテーブルが含まれます。

- ネットワークまたはメッセージバッファ
- ストリーム バッファ
- I/O バッファ キャッシュ

メッセージ バッファグループ

メッセージバッファがどのように使用されるかをモニタするには、メッセージ バッファグループを使用します。

クラス：mbufs

このグループには、メッセージバッファタイプごとの割り当てを追跡するメッセージバッファ割り当てテーブルも含まれます。

クラス：mbufAllocEntry

インスタンス：特定のメッセージバッファタイプの `mbufDesc` 属性値

ストリーム バッファ グループ

このテーブルは、エージェントが存在するノードの **WLM** クラスを定義します。このテーブル内のデータは、**SystemEDGE** エージェントが **AIX** ベースの **OS** 上で実行されており、**OS** 内のワーク ロード マネージャが有効な場合にのみ利用可能です。

ストリーム バッファ割り当てテーブル

ストリーム サブシステムが使用しているバッファのバッファ割り当ておよび使用率の統計をモニタするには、ストリーム バッファ割り当てテーブルを使用します。

クラス : `strbufAllocEntry`

インスタンス : 特定のストリーム バッファの `strbufAllocIndex` 属性値

I/O バッファ キャッシュ グループ

ベーシック ディスク I/O の I/O バッファ割り当ておよび使用率を追跡するには、I/O バッファ キャッシュ グループを使用します。

クラス : `ioBufferCache`

これらのカウンタの値をグラフ化して、I/O バッファ アクティビティのピーク時を検出することもできます。

ディレクトリ名のルックアップ キャッシュ グループ

Unix でディレクトリのキャッシュおよびファイル名の **vnode** へのマッピングをモニタするには、ディレクトリ名のルックアップ キャッシュ (**DNLC**) グループを使用します。このグループには、このキャッシュのパフォーマンス統計が含まれます。

クラス : `dnlc`

AIX 論理パーティション グループ

AIX 論理パーティション グループでは、AIX システム上の各論理パーティション (LPAR) に関する情報をモニタできます。

クラス : logicalPartition

AIX 論理パーティション グループを使用して LPAR 環境をモニタする方法の詳細 (MIB の各属性の説明を含む) については、AIX 論理パーティション テーブルを参照してください。

ワークロード マネージャ(WLM)クラスのテーブル

トラップ コミュニティ テーブルでは、現在設定されているアクセス コミュニティやその他の SNMP 固有の設定情報を確認できます。CA Virtual Assurance ユーザ インターフェースから、このテーブル内の設定を編集できます。テーブルには、実行されているバージョンに応じて、サポートしている SNMPv1、SNMPv2、および SNMPv3 の情報が含まれます。

クラス : trapCommunityEntry

インスタンス : 特定のトラップ送信先とトラップ ポートの tcDestinationAddr 属性値および tcDestinationPort 属性値

トラップ コミュニティ テーブル

トラップ コミュニティ テーブルでは、現在設定されているアクセス コミュニティやその他の SNMP 固有の設定情報を確認できます。CA Virtual Assurance ユーザ インターフェースから、このテーブル内の設定を編集できます。テーブルには、実行されているバージョンに応じて、サポートしている SNMPv1、SNMPv2、および SNMPv3 の情報が含まれます。

クラス : trapCommunityEntry

インスタンス : 特定のトラップ送信先とトラップ ポートの tcDestinationAddr 属性値および tcDestinationPort 属性値

NT システム グループ

グループとテーブルが含まれる NT システム グループは、Windows システム用に設計されています。

クラス : ntSystem

注: 通常、SystemEDGE エージェントでは、Windows と UNIX に対して同じ MIB オブジェクトをサポートしています。ただし、基盤となる Windows オペレーティングシステムでシステム管理 MIB のオブジェクトが一部サポートされていません。Windows 上でサポートされていない MIB オブジェクトのリストについては、「SystemEDGE リリース ノート」を参照してください。

NT スレッド テーブル

Windows オペレーティングシステムでは、プロセッサ上で実行するスレッドがカーネルによってスケジュールされます。スレッドは、プログラム コードを実行するプロセスの一部です。1つのプロセスに複数のスレッドが含まれる場合もあります。

システム上で実行されている各スレッドの詳細なステータス情報を取得するには、NT スレッド テーブルを使用します。

クラス : ntThreadEntry

インスタンス : 特定のスレッドの ntThreadPID 属性値および ntThreadNumber 属性値

NT レジストリ グループ

Windows レジストリは、ハードウェア、ユーザ アカウント、アプリケーションなどのシステム設定の詳細を格納するデータベース リポジトリとして機能します。Windows レジストリを照会するには、NT レジストリ グループを使用します。

クラス : ntRegistry

以下の情報を照会することにより、レジストリが設定されたサイズ制限を超えて増大する危険性があるかどうかを判断できます。

- レジストリの現在のサイズ (MB) (ntRegistryCurrentSize)
- レジストリのサイズ制限 (MB) (ntRegistrySizeLimit)

NT サービス テーブル

Windows には、Windows サービスとして実行されるいくつかのプログラムがあります。Windows サービスをモニタするには、NT サービス テーブルを使用します。

クラス : ntServiceEntry

インスタンス : 特定のサービスの ntServiceName 属性値

NT パフォーマンス グループ

NT パフォーマンス グループは、以下の Windows パフォーマンス メトリックをモニタするために用意されています。

- システム
- キャッシュ
- メモリ
- ページファイル

NT システム パフォーマンス グループ

NT システム パフォーマンス グループの統計を使用すると、システムの Windows オペレーティング システムのヘルスおよびパフォーマンスを追跡できます。

クラス : ntSystemPerf

たとえば、通常、過負荷状態のシステムには高いレベルのシステム アクティビティがあり、多くのプロセス スレッドが CPU 時間をめぐって競合し、複数のジョブがキューで実行を待機しています。ntContextSwitches および ntRunQLen の MIB オブジェクトをモニタすることにより、システム アクティビティのレベルを追跡できます。

ntContextSwitches では、発生したコンテキスト切り替え数をカウントします。コンテキスト切り替えは、あるスレッドが CPU を解放し、別のスレッドが CPU を取得するたびに発生します。コンテキスト切り替え率が高い場合は、システムの負荷が高いことを示しています。ntRunQLen は、実行を待機中のスレッドのバックアップがあるかどうかを示します。

NT キャッシュ パフォーマンス グループ

NT キャッシュ パフォーマンス グループには、システム全体のバッファおよびファイル システム キャッシュの統計が含まれているので、システムのキャッシュ メカニズムの有効性を判断できます。

クラス : ntCachePerf

アプリケーションがデータを要求すると、データはまずキャッシュにマップされ、次にキャッシュからメモリにコピーされます。その後、アプリケーションによって変更されたデータが、**Lazy Writer** システム スレッド、またはアプリケーションからのライトスルー要求によってキャッシュからディスクに書き込まれます。

NT メモリ パフォーマンス グループ

NT メモリ パフォーマンス グループには、メモリとページングのパフォーマンス カウンタおよび統計が含まれます。システムのメモリ使用率を追跡するには、このグループを使用します。

クラス : ntMemoryPerf

NT ページ ファイル パフォーマンス グループ

Windows では、**Pagefile.sys** ページング ファイルを使用して、メモリの追加要求を処理します。ページング ファイルは、オペレーティング システムによって予約されているディスク領域のブロックであり、メモリ マネージャが必要に応じて領域を作成できます。

システムにメモリを追加できるかどうかを判断するには、**NT ページ ファイル パフォーマンス グループ**を使用して、ページング ファイルのサイズをモニタします。

クラス : ntPageFilePerf

NT イベント モニタ グループ

Windows では、イベント ログを使用して、重要なシステムおよびアプリケーションのステータス メッセージをキャプチャします。これらのイベント ログへは、**NT イベント モニタ グループ**からアクセスできます。

クラス : ntEvents

NT イベント モニタ テーブルを使用すれば、エージェントに対して、指定したイベントの **Windows** イベント ログを継続的にモニタするように指示できます。一致するイベントがシステム上で生成されるたびに、エージェントから管理システムにトラップ メッセージが通知されます。アクション コマンドを実行して、イベントをただちに処理することもできます。NT イベント テーブルは、各モニタ エントリの詳細なステータス情報を提供します。

クラス : ntEventMonEntry

インスタンス : 特定のモニタ エントリの ntEventMonDescr 属性値

NT イベント モニタの詳細については、「**Windows イベント モニタリング**」の章を参照してください。

NT イベント ログ モニタリング テーブル

このテーブルには **NT** イベント ログ モニタ エントリが含まれます。各エントリは、イベント ログと正規表現を指定します。エージェントは、イベントが更新されたかどうかを確認するために定期的にイベント ログを取ります。イベントが更新されている場合、エージェントは正規表現の新しいエントリをスキャンします。エントリが正規表現に一致すると、トラップ PDU が適切なマネージャに送信されます。

NT レジストリ/パフォーマンス拡張グループ

SystemEDGE エージェントは、**Windows** レジストリおよびパフォーマンス カウンタの情報を取り込んで、**Systems Management Empire MIB** を拡張する強力なメカニズムを提供します。この情報には、設定データ（通常は **regedit** を使用して表示）およびパフォーマンス データ（通常は **perfmon** を使用して表示）の両方が含まれます。

この機能を使用すると、エージェントをカスタマイズして、システムとアプリケーションに関する追加の設定情報およびパフォーマンス情報を返すことができます。たとえば、多くのアプリケーションは、アプリケーションの設定を指定するレジストリ エントリを提供するので、エージェントはこれらのエントリを **SNMP** 経由で提供できます。**SystemEDGE** エージェントは、多くのアプリケーションが提供している統計にもアクセスできます。

この機能は NT レジストリおよびパフォーマンス グループでサポートされます。

クラス : ntRegPerf

このグループには、ユーザが設定可能な未定義のスカラ MIB 変数が 128 個あります。これらの変数に対する SNMP Get 要求への応答として、SystemEDGE エージェントは Windows レジストリを読み取って値を返します。

NT レジストリおよびパフォーマンス グループで MIB オブジェクトを使用および設定する方法の詳細については、「Windows レジストリとパフォーマンス MIB オブジェクト」の章を参照してください。

セルフ モニタ テーブル

管理対象の整数ベースの MIB 変数をモニタするためにエージェントのセルフ モニタリング機能を設定するには、セルフ モニタ テーブルを使用します。エージェントは自動的に MIB 変数をモニタして、しきい値違反が発生した場合に状態変更トラップで管理システムに通知します。セルフ モニタ テーブルは、テーブルによって各モニタのステータスが保持されるステートフルモニタリングをサポートし、その他のテーブルのセルフ モニタを集計して、オブジェクト状態を計算します。

クラス : monitorEntry

インスタンス : 特定のセルフ モニタ エントリの monDescr 属性値

セルフ モニタ テーブルおよび MIB オブジェクトのモニタリングの詳細については、「セルフ モニタリング」の章を参照してください。

分散システム グループ

分散システム グループは、すべてのプラットフォーム上で以下の分散システム機能をモニタおよび管理するために必要なオブジェクトを提供します。

- RPC
- NFS

RPC 統計グループ

カーネルの **RPC**（リモートプロシージャコール）機能の使用に関連する統計およびカウンタを追跡するには、**RPC 統計グループ**を使用します。

クラス： rpc

これらのカウンタの値をグラフ化すると、**RPC** アクティビティのピーク時を検出できます。統計とカウンタは、クライアント側およびサーバ側のプロトコル処理に対する適応性に基づいて分類されます。

注：RPC の詳細については、「**RFC 1057**」を参照してください。

NFS 統計グループ

カーネルの **NFS** 機能の使用に関連する統計およびカウンタを追跡するには、**NFS 統計グループ**を使用します。

クラス： nfs

これらのカウンタの値をグラフ化すると、**NFS** アクティビティのピーク時を検出できます。統計とカウンタは、クライアント側およびサーバ側のプロトコル処理に対する適応性に基づいて分類されます。

注：NFS の詳細については、「**RFC 1094**」を参照してください。

履歴テーブル

SystemEDGE は、後で取得できるように、時間の経過と共に整数ベースの各種 MIB オブジェクト（カウンタ、ゲージなど）の値を追跡して、それらの値を格納します。この機能は、一般に履歴サンプリングと呼ばれており、ネットワーク全体の管理ステーションのポーリング量を削減します。エージェントは 2 つの SNMP MIB テーブルを介してこの機能を提供します。

履歴制御テーブル

クラス： `empireHistoryCtrlEntry`

インスタンス：特定の履歴制御テーブルエントリの `empireHistoryCtrlDescr` 属性値

履歴テーブル

クラス： `empireHistoryEntry`

インスタンス：特定の履歴テーブルエントリの `empireHistorySampleIndex` 属性値

履歴制御テーブルには、サンプリングの対象であり、履歴テーブルに保存されるデータを表すパラメータが用意されています。制御テーブルの各行は、テーブルのパラメータ（列オブジェクト）に値を割り当てることにより、特定のデータ収集関数を定義します。履歴テーブル内の 1 つ以上の行（格納されたサンプル）が、その 1 つの制御行と関連付けられます。

制御テーブルの各行には、一意のインデックス値（`empireHistoryCtrlIndex`）が割り当てられます。各行にサンプリング対象のオブジェクトインスタンスを指定することにより、サンプリングの頻度（30 秒の倍数単位）や保持するサンプルの数（バケット）など、データ収集関数が定義されます。データ収集関数（制御テーブルの行）ごとに、履歴テーブルの行セットが関連付けられます。履歴テーブルの各行（別名：バケット）には、1 回のサンプリング間隔で収集された、特定の MIB オブジェクトの値が保持されます。

サンプリング間隔が発生するたびに、エージェントによって履歴テーブルに行が追加され、このデータ収集関数に関連付けられている行セットと同じ `empireHistoryIndex` 値が挿入されます。この新規行は履歴制御テーブルの 1 行に対応し、`empireHistorySampleIndex` 値は、直前のサンプルの `SampleIndex` 値よりも 1 つ大きくなります。

これらのテーブルの属性、およびその属性を履歴の収集で使用方法の詳細については、「履歴の収集」の章を参照してください。

ログ モニタ テーブル

ログ モニタ テーブルを使用すると、エージェントを設定して、システム ログ ファイルを正規表現に照らしてモニタできます。また、ファイルディレクトリのサイズと内容もモニタできます。

クラス : `logMonitorEntry`

インスタンス : 特定のログ モニタ エントリの `logMonitorLogFile` 属性値

たとえば、`/var/adm/sulog` ログ ファイルで正規表現「`su.*fail`」を検索するようにエージェントを設定できます。一致するエントリが検出されるたびに、設定された管理システムに対して、エージェントから **SNMP** トラップが送信されます。

ログ モニタ テーブルの各行は、特定の正規表現によるログ ファイルのモニタリングを表します。

ログ モニタ テーブルの属性、およびこのテーブルを使用してログ ファイルをモニタする方法の詳細については、「ログ ファイル モニタリング」の章を参照してください。

ディスク統計テーブル

ディスク統計テーブルは、ディスクの I/O 統計を提供します。

クラス : `diskStatsEntry`

インスタンス : 特定のディスク ドライブの `diskStatsHostmibDevTableIndex` 属性値

各テーブルエントリは、1 つのディスクに関する最新のディスク統計を提供します。エージェントは、定期的（60 秒ごと）に各ディスクのシステム データ構造のステータスを確認し、その値をテーブルに記録します。

ディスク統計テーブルを使用するときは、`diskStatsLastUpdate` 列を参照して、この行が最後に更新された時間（`TimeTick`）を確認してください。この列では、最後にテーブルを照会した後で統計が更新されているかどうかを把握できます。

AIX システムのディスク パフォーマンス統計収集の設定

AIX システムでディスク パフォーマンス統計を収集するには、システムを手動で設定する必要があります。オペレーティングシステムに対して、この統計を追跡するように指示した場合にのみ、SystemEDGE エージェントはこれらの統計をモニタできます。

注: これらの統計収集を有効にすると、ディスクのスループットが減少します。

AIX システムのディスク統計収集を有効にする方法

1. モニタするシステムにルート権限でログインします。
2. コマンドプロンプトで次のコマンドを入力します。

```
chdev -l sys0 -a iostat=true
```

AIX システムのディスク統計収集を無効にする方法

1. モニタするシステムにルート権限でログインします。
2. コマンドプロンプトで次のコマンドを入力します。

```
chdev -l sys0 -a iostat=false
```

CPU 統計テーブル

CPU 統計テーブルは、各 CPU のパフォーマンス統計を提供します。

クラス : `cpuStatsEntry`

インスタンス : 特定の CPU の `cpuStatsDescr` 属性値および `cpuStatsIndex` 属性値

各テーブルエントリは、1 つの CPU に関する最新の統計を提供します。エージェントは、定期的（60 秒ごと）に各 CPU のシステムデータ構造のステータスを確認し、その値をテーブルに記録します。以下の統計をモニタするには、CPU 統計テーブルを使用します。

- CPU タイプ (`cpuStatsDesc`)
- アイドル、ユーザ、システム、または待機モードで動作した時間 (`cpuStatsIdle`、`cpuStatsUser`、`cpuStatsSys`、`cpuStatsWait`)
- 統計が最後に更新された時間 (`cpuStatsLastUpdate`)
- 論理 CPU が消費した物理プロセッサの割合 (`cpuStatsPC`)
- 論理 CPU が消費した獲得可能容量の割合 (`cpuStatsEC`)
- アイドル、ユーザ、システム、または待機モードで動作した時間の割合 (`cpuStatsIdlePercent`、`cpuStatsUserPercent`、`cpuStatsSysPercent`、`cpuStatsWaitPercent`)

CPU 統計テーブルを使用するときは、`cpuStatsLastUpdate` 列を参照して、この行が最後に更新された時間 (`TimeTick`) を確認してください。この列では、最後にテーブルを照会した後で統計が更新されているかどうかを把握できます。

クラス : `cpuGroup`

各テーブルエントリは、1 つのグループ内にあるすべての CPU に関する最新の統計を提供します。関連する CPU の合計グループでは、テーブルからメトリックを取得して、すべての CPU の合計を表示します。また、LPAR が消費したプロセッサの合計および獲得可能容量の合計も表示されます。

拡張グループ

拡張グループは、システムおよびアプリケーションに関する広範な情報を取り込んで、Systems Management Empire MIB を拡張する強力なメカニズムを提供します。

クラス : extensionGroup

このグループには、ユーザが設定可能な未定義のスカラ MIB 変数が 232 個あります。これらの変数に対する SNMP Get 要求への応答として、SystemEDGE エージェントは、この変数に指定されているコマンドを呼び出して、そのコマンドから返された値を返します。また、SNMP Set 操作を使用すると、パラメータをコマンドへ渡すこともできます。

拡張グループ変数の使用の詳細については、「カスタム MIB オブジェクト」の章を参照してください。

プロセス モニタ テーブル

プロセス モニタ テーブルは、アプリケーションとプロセスをモニタするための強力な柔軟なメカニズムを提供します。主要プロセスとアプリケーションのさまざまな属性をモニタして、一定のしきい値を超えたときに SNMP 状態変更トラップを送信できます。セルフ モニタ テーブルと同様に、プロセス モニタ テーブルはステートフル（処理状態を把握する）モニタリングをサポートします。

クラス : processMonEntry

インスタンス : 特定のプロセスの pmonRegExpr 属性値

プロセス モニタ テーブルの属性およびプロセスのモニタリングの詳細については、「プロセスとサービスのモニタリング」の章を参照してください。

[プロセス グループ モニタ]テーブル

プロセス グループ モニタ テーブルは、プロセス グループをモニタするための強力で柔軟なメカニズムを提供します。正規表現を使用して、プロセス グループ モニタ テーブルにエントリを作成できます。テーブルでは、正規表現に一致するプロセスの統計を合計し、グループのグループ変更をモニタします。

クラス : `processGroupMonEntry`

インスタンス : 特定のプロセス グループの `pgmonProcRegExpr` 属性値

プロセス グループ モニタ テーブルの属性およびプロセス グループのモニタリングの詳細については、「プロセス グループ モニタリング」の章を参照してください。

集計状態テーブル

集計状態テーブルでは、同じクラス、インスタンス、および属性値を持つセルフモニタとプロセス モニタを 1 つのテーブル エントリに集計して、集計オブジェクトの状態を保持します。このテーブルを使用すると、エージェントはセルフモニタ テーブルおよびプロセス モニタ テーブル内のエントリのステータス情報を保持できます。

クラス : `aggregateEntry`

インスタンス : 特定のテーブル エントリの `aggObjInstance` 属性値および `aggObjAttribute` 属性値

注: エージェントを設定して、集計レベルを上げることができます。詳細については、「エージェントの設定」の章を参照してください。

LPAR グループ

このグループは、論理パーティション (LPAR) の物理 CPU および論理 CPU の統計を提供します。

クラス : `totalPhysicalCpuStats`

クラス : `logicalPartition`

ミラー テーブル

Systems Management Empire MIB には、集計テーブルおよびセルフ モニタ テーブルのミラー テーブルが用意されています。これらのテーブルはメインテーブルと同じ属性およびエントリを備え、メインテーブルに対する相互参照として使用されます。

自動ウォッチャー テーブル

自動ウォッチャー テーブルには、エージェントの自動モニタリング アクティビティが記述されています。このテーブルの行により、エージェントは、指定された条件に基づいて、すべての一致するインスタンスのモニタ エントリを自動的に作成します。

サポートされていない Systems Management Empire MIB オブジェクト

SystemEDGE は、サポートされているすべてのオペレーティング システム上の Systems Management Empire MIB をサポートします。しかし、この MIB モジュール内の一部の MIB オブジェクトは基盤となるオペレーティング システム上でサポートされていません。さらに、UNIX に固有の MIB オブジェクトの一部は Windows に適用できません。そのため、Windows バージョンのエージェントを使用して、それらのオブジェクトを実装することはできません。

すべてのオペレーティング システム上のサポートされていない Systems Management Empire MIB オブジェクトのリストについては、「CA SystemEDGE リリース ノート」を参照してください。

第 8 章：セルフ モニタリング

この章では、SystemEDGE エージェントを使用してユーザ指定のしきい値と突き合わせて MIB 変数をモニタする方法について説明します。

このセクションには、以下のトピックが含まれています。

[セルフ モニタリングの概要](#) (P. 231)

[セルフ モニタの状態管理](#) (P. 232)

[セルフ モニタ テーブル](#) (P. 235)

[セルフ モニタリングの設定](#) (P. 252)

[セルフ モニタリング用の edgemon コマンド](#) (P. 278)

[セルフ モニタリング エントリの削除](#) (P. 283)

セルフ モニタリングの概要

SystemEDGE エージェントでは、その管理の下でどんな整数ベースの MIB 変数も動的にモニタできます。セルフ モニタ テーブルを使用して、モニタする変数、ポーリング間隔、比較演算子（次の値より大きい、次の値と等しいなど）、しきい値、重大度、オブジェクト情報、および他の値を指定します。SystemEDGE エージェントは自動的にその変数をモニタし、指定した条件が満たされる場合には管理システムにトラップを送信します。

注：モニタで状態モデルが使用される場合、SystemEDGE は、定義された各モニタに対してではなく、管理対象オブジェクトに対してトラップを送信します。

SystemEDGE エージェントは、指定されたあらゆるセルフ モニタの状態値を管理および更新し、オブジェクトクラス、インスタンス、および属性値を使用して、複数の定義済みモニタでオブジェクトのステータスを集計することができます。

エージェントはまた、管理対象システム上で管理機能を果たすコマンドを実行して、例外の原因をすぐに修正します。たとえば、ルート ファイルシステムの容量の比率をモニタし、容量不足になったらマネージャに通知するようにエージェントを設定できます。

セルフ モニタの状態管理

SystemEDGE 状態管理モデルは、セルフ モニタ テーブル内のすべてのエントリの状態値を管理し、ユーザがオブジェクト全体の状態を取得するために管理対象オブジェクトを定義できるようにします。状態管理モデルは、セルフ モニタ テーブル内のエントリに対して以下の機能を提供します。

- エージェントが各エントリの状態値を管理するために使用する、各セルフ モニタ エントリの重大度を定義する機能。重大度は CIM に準拠した状態に変換されます。
- オブジェクトクラス、インスタンス、および属性値を使用して、3 段階ネーミング スキームによって 1 つ以上の管理対象オブジェクトに 1 つ以上のモニタされたリソースをグループ化する機能。このメカニズムは柔軟であるため、ユーザはさまざまな機能領域のリソースを動的にグループ化できます。
- 特定のオブジェクトの状態が階層のより高いレベルのオブジェクトにプロパゲートされる、スーパー管理対象オブジェクトの階層を作成する機能。
- 管理対象オブジェクト全体の状態が変化した場合の状態変更トラップによる通知

状態管理モデルは、CA Virtual Assurance オブジェクト モデルに統合されます。AOM データベース内の管理対象オブジェクトはそれぞれ CIM に準拠した状態になり、SystemEDGE 管理対象オブジェクトおよび状態は SystemEDGE プラットフォーム管理モジュールによって CA Virtual Assurance に反映されます。CA Virtual Assurance によって SystemEDGE を管理すると、CA Virtual Assurance ユーザ インターフェース内の管理対象オブジェクトの状態情報を視覚化し、CA Virtual Assurance Event ビューを通して状態変更トラップを確認することができます。

CA Virtual Assurance 内での SystemEDGE 管理対象オブジェクトの視覚化の詳細については、CA Virtual Assurance のドキュメントを参照してください。

状態管理モデルを利用するためのセルフ モニタの設定の詳細（属性値および構文例を含む）については、この章の残りで説明します。

管理対象オブジェクトの作成

オブジェクトクラス、インスタンス、および属性値を使用して、セルフ モニタ テーブルから管理対象オブジェクトの作成を制御します。デフォルトでは、SystemEDGE は、各一意のクラス、インスタンス、および属性セルフ モニタの管理対象オブジェクトを作成します。複数のセルフ モニタでこれらの 3 つのプロパティに対して同じ値が含まれる場合、エージェントは単一の管理対象オブジェクトにそれらをグループ化し、オブジェクト全体の状態を取得します。

ユーザがどのようにこのメカニズムを使用できるかの柔軟性を例証するには、以下のシナリオを検討します。

- CPU 使用率をモニタするために、クラス、インスタンス、および属性値が同じ 3 つのセルフ モニタ エントリを定義します。モニタは、85% を超える CPU 使用率に対して **warning** 重大度、90% を超える CPU 使用率に対して **major** 重大度、および 95% を超える CPU 使用率に対して **critical** 重大度を定義します。エージェントは、これらのモニタを 1 つの管理対象オブジェクトに組み合わせ、現在の最悪の重大度をオブジェクト状態として管理します。
- ユーザの企業で接続されるさまざまな機能領域からメトリックをモニタするために、クラス、インスタンス、および属性値が同じ 3 つのセルフ モニタ エントリを定義します。たとえば、基本ディスク領域のセルフ モニタ、SRM AIM からのレスポンス時間モニタ、およびリモート モニタリング AIM からのファイルシステム モニタを組み合わせ、各セルフ モニタに **warning** 重大度を割り当てることができます。このシナリオでは、3 つのセルフ モニタしきい値 (AND フラグ) すべてに違反する場合のみ、エージェントは警告状態変更トラップを送信します。
- オブジェクト インスタンスまたは属性に対するオブジェクト クラスとアスタリスク値が同じ値のセルフ モニタ エントリのスーパー集計を定義します。このシナリオは、それぞれの多数のオブジェクトをグループ化します。たとえば、トップレベルのシステム クラスおよび「**Filesystem**」インスタンスを備えたすべてのオブジェクトを集計できます。その結果、すべてのファイルシステムのセルフ モニタは現在の最悪の状態をトップレベル オブジェクトにプロパゲートします。

注: より高いレベルのすべてのオブジェクトを集計するようにエージェントを設定することもできます。詳細については、「エージェントの設定」の章を参照してください。

これらのシナリオの構文例については、「[セルフ モニタリングの例](#) (P. 260)」を参照してください。

ユーザがオブジェクト クラス、インスタンス、および属性値を定義しない場合、SystemEDGE は、エージェントインストールの設定ディレクトリ内にある `sysedge.oid` ファイルを使用して、これらのプロパティのデフォルト値を割り当てます。このファイルは、エージェントでサポートされているすべての MIB 内のすべての属性のデフォルト クラスおよび属性を定義します。このファイルは、[MIB テーブル](#) (P. 45)内の各 MIB 属性のデフォルト インスタンスも定義します。デフォルトの割り当てでは、同じ MIB 属性をモニタしているセルフ モニタのオブジェクトをエージェントが引き続き作成します。

Systems Management Empire MIB 内のグループ、テーブルおよび属性のデフォルト割り当ての詳細については、「Systems Management Empire MIB」を参照してください。

関連項目

[オブジェクト ステータス](#) (P. 71)

[オブジェクト集計](#) (P. 72)

[スーパー集計](#) (P. 74)

[状態管理と集計の仕組み](#) (P. 76)

集計状態テーブル

集計状態テーブルは、管理対象オブジェクト表現を作成し、オブジェクト全体の状態を取得するために必要な機能を果たします。集計状態テーブルは、**SystemEDGE** セルフ モニタおよびプロセス モニタのフィルタされたビューです。テーブルには、管理対象オブジェクトの名前付けおよび各モニタ エントリによって指定された状態情報に基づいて計算された、集計状態情報が含まれます。

テーブルは、オブジェクト クラス、インスタンス、および属性値（またはより高いレベルでグループ化されたオブジェクト）が同じすべてのセルフ モニタおよびプロセス モニタの集計状態テーブル エントリを作成します。セルフ モニタとプロセス モニタは、セルフ モニタ テーブルおよびプロセス モニタ テーブル内に個別のエントリとしてのみ存在します。集計状態テーブルは、これらのモニタのグループ化を管理し、テーブルを連続的にポーリングし、管理対象オブジェクトの状態が変化したときに状態変更トラップを送信します。

注: 新しい管理対象オブジェクトが作成されるか、既存の管理対象オブジェクトが削除されたときも、集計状態テーブルはトラップを送信します。

集計状態テーブルがどのように管理対象オブジェクトとそれらの状態を管理するかの詳細については、「コンセプト」の章を参照してください。テーブル列の説明については、「集計状態テーブル」を参照してください。

セルフ モニタ テーブル

Systems Management Empire MIB のセルフ モニタ テーブルは、エージェントが現在モニタしている各 MIB 属性についての情報を提供します。行はそれぞれ、定義されたセルフ モニタ エントリを表します。

エントリごとに、テーブルは以下の種類の情報を提供します。

- インデックス
- 説明
- エージェントが変数をポーリングする間隔
- サンプル タイプ（絶対値またはデルタ値）
- オブジェクト ID
- エージェントがモニタしている現在の値
- しきい値
- 最後の試呼
- すでに送信されたトラップの数
- 最後のトラップ
- 行ステータス
- サンプリングされた最小値と最大値
- トラップが送信されるときに講ずるべき処置
- フラグ
- モニタのインデックスによる先行
- オブジェクト クラス
- インスタンス
- 属性
- 定義された重大度
- 現在のモニタ ステータス

セルフ モニタ テーブルの列

以下のリストは、セルフ モニタ テーブルの列を定義します。セルフ モニタ テーブルおよびそのフィールドの詳細については、エージェントインストールの `mib` サブディレクトリ内にある、`empire.asn1` ファイルを参照してください。

`monIndex`

このエントリの一意の行インデックスを定義します (**MAXINT** に対しては **11**)。行 **1** ~ **10** はエージェントの内部使用に予約済みです。

権限：読み取り専用

`monDesc`

モニタされているオブジェクトの説明 (長さが **0** ~ **512** 文字) を定義します。

権限：読み書き

デフォルト：デフォルトのエントリ

`monInterval`

数秒間にエージェントが変数をどれくらいの頻度でポーリング (モニタ) するかを定義します。値は **30** 秒以上かつ **30** 秒の倍数にする必要があります。

権限：読み書き

デフォルト： **60**

`monSampleType`

エージェントがオブジェクトの絶対値 (**absoluteValue(1)**) をサンプリングするか、連続するサンプル (**deltaValue(2)**) 間の差異を取得するかを定義します。たとえば、カウンタ変数をモニタするには **deltaValue** を使用します。**deltaValue** は変化率を表すからです。ゲージをモニタするには、**absoluteValue** を使用します。**absoluteValue** はオブジェクトの正確な値を表すからです。

権限：読み書き

デフォルト： **absoluteValue (1)**

monOID

モニタする **MIB** オブジェクトを表す完全なオブジェクト識別子を定義します。オブジェクト識別子のインスタンス部分（たとえばスカラの場合は **.0**）が必要です。オブジェクトインスタンスが存在する必要があり、**SystemEDGE** エージェントでサポートされている **MIB** 内に含まれている必要があります。オブジェクトは整数タイプ（カウンタ、ゲージ、タイム チックなど）である必要があります。

権限：読み書き

monCurrVal

モニタされている **MIB** 変数の最後にサンプリングされた値を指定します。**monInterval** 秒ごとに、エージェントは変数の最新の値を反映するためにこのフィールドを更新します。

権限：読み取り専用

monOperator

表現 *curval* 演算子の値を評価するために使用されるブール演算子を定義します。演算子には、以下のいずれかの値を指定できます。

- **nop(1)**（演算なし。オブジェクトの値を取得するが、ブール式を評価しません。）
- **gt(2)**（次の値より大きい）
- **lt(3)**（次の値より小さい）
- **ge(4)**（次の値以上）
- **le(5)**（次の値以下）
- **eq(6)**（等しい）
- **ne(7)**（等しくない）

権限：読み書き

デフォルト：**nop(1)**

monValue

各モニタリング サイクル中にモニタされる MIB 変数の現在の値をエージェントが **monOperator** を使用して比較する整数値を定義します。比較が **true** として評価された場合、エージェントはモニタ状態を変更するか（重大度が **none** 以外のセルフ モニタの場合）、しきい値違反トラップを送信します（重大度が **none** のセルフ モニタの場合）。

たとえば、ゲージの値が **100** を超える場合に通知されるようにする場合は、**monValue** として **100** を設定します。

権限：読み書き

monLastCall

エージェントがモニタしている MIB 変数を最後にサンプリングした時間を（**sysUpTime** に基づいて）定義します。**0** は、MIB 変数がまだサンプリングされていないことを示します。

権限：読み取り専用

monNumTraps

エージェントがこのエントリに送信したトラップの数を定義します。この値は **monitorTrap** を追跡します。**monitorTrap** は、重大度が **none** のセルフ モニタ（レガシー モニタ）でしきい値違反が発生したときにエージェントによって送信されます。この値は、レガシー モニタに対して例外条件が発生している頻度を決定するのに役立ちます。この値は、見つからないトラップメッセージを検出する手段も提供します。

権限：読み取り専用

注：ステートフルセルフ モニタの管理対象オブジェクトに対して送信された状態変更トラップの数の詳細については、「集計状態テーブル」を参照してください。

monLastTrap

エージェントがこのエントリに最後にトラップを送信した時間を（**sysUpTime** に基づいて）定義します。この値は **monitorTrap** を追跡します。**monitorTrap** は、重大度が **none** のセルフ モニタでしきい値違反が発生したときにエージェントによって送信されます。

権限：読み取り専用

注：ステートフルセルフ モニタの管理対象オブジェクトに対して送信された最後の状態変更トラップの詳細については、「集計状態テーブル」を参照してください。

monRowStatus

行ステータスを定義します。以下のいずれかになります。

- active
- notInService
- notReady
- createAndGo
- createAndWait
- destroy(6)

通常、行は **active** または **notInService** のいずれかです。これらの値は、**SNMPv2** の **SMI RowStatus** テキスト表記法によって定義された値と意味が同一です。

権限：読み書き

デフォルト：**createAndWait(5)**

monMinValue

エージェントが **MIB** 変数のポーリングを開始してから観察した最低（最小）値を定義します。

権限：読み取り専用

monMaxValue

エージェントが **MIB** 変数のポーリングを開始してから観察した最高（最大）値を指定します。

権限：読み取り専用

monAction

表現が **True** として評価され、エージェントがしきい値違反トラップ（レガシー モニタ）または状態変更トラップ（ステートフル モニタ）を送信したときに実行される引用符で囲まれたコマンド（長さが **0** ～ **2048** 文字）を、任意のパラメータと共に定義します。ステートフルセルフ モニタについては、しきい値違反がオブジェクトの状態変更をもたらさない場合、コマンドは実行されません。文字列が空の場合、エージェントはこのエントリのアクションを実行しません。

注:アクションに **Windows** バッチ ファイルを使用しないでください。プログラム上の厳しい制限があり、デスクトップアプリケーションでは正しく動作しないことがよくあります。代わりに、**Perl** または **Visual Basic** など、強力な柔軟なスクリプト言語を使用してください。

権限：読み書き

monFlags

エントリが操作中に従う追加動作セマンティックを示す整数フラグを定義します。利用可能なフラグおよびフラグ設定の詳細については、「[モニタ テーブル フラグ \(P. 245\)](#)」を参照してください。

権限：読み書き

monSupersededBy

このエントリに先行するモニタのインデックスを指定します。先行するモニタはテーブルエントリの前に評価されます。先行するモニタのしきい値を超えている場合、このエントリはそれ以上評価されなくなります。このような関連するモニタのシーケンスによって、トラップの不要なシーケンスを排除することができます。値がゼロ (0) の場合、モニタ エントリの関連付けは行われなことを示します。

注: オブジェクト状態モデルを使用する場合は、この機能は古いものとして無効になります。

権限：読み書き

monObjClass

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクトクラスを定義します。エージェントは、**sysedge.oid** ファイルを使用してデフォルト値を割り当てます。クラス名はモニタされたオブジェクト テーブルを指定するために使用できます。**SystemEDGE** が指定された名前を持つ既存のテーブルを決定しない場合、**SystemEDGE** はクラスを作成します。クラスはモニタリング オブジェクト識別子 (SNMP OID) を指定し、集計目的で参照できます。

重要: 新たに作成された参照は、モニタリング エントリが既存の SNMP OID に解決される場合のみ有効です。

monObjInstance

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト インスタンスを定義します。値が設定されていない場合、エージェントでデフォルトを判断できなければ、値は「不明」になります。表形式以外のオブジェクトのデフォルトは「NULL」です。リモート オブジェクトのシームレスなモニタリングの場合、インスタンス名はプレフィックス `//hostname/` で開始されます。

ローカル オブジェクトの場合は、エージェントがプレフィックス `//.` を各インスタンス名に追加します（「NULL」を含む）。この属性は、空文字列または「*」に設定することはできません。そのような要求は無視されます。インスタンス名が対応するインスタンスへの解析時に一致できない場合、**SystemEDGE** はこのモニタリング エントリの **OID** を指定されたインスタンス名で作成します。

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

monObjAttribute

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト インスタンスを定義します。エージェントは、`sysedge.oid` ファイルを使用して、MIB 属性名に対応するオブジェクトのデフォルト値を割り当てます。属性名がモニタリング エントリの解析時に不明な場合、**SystemEDGE** は指定された名前属性を作成します。

権限：読み書き

デフォルト：モニタされる **OID** の **ASCII MIB 属性名**

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

monSeverity

オブジェクト状態管理モデルに使用する重大度を定義します。この値は、しきい値違反が発生したときにエントリに割り当てる状態を定義します。有効な値は以下のとおりです。

- none(1)
- ok(2)
- warning(3)
- minor(4)
- major(5)
- critical(6)
- fatal(7)

none の値はレガシー モニタを作成し、エントリがオブジェクト状態管理モデルから除外されます。

権限：読み書き

デフォルト：none

monCurrState

エントリの取得された現在の状態を定義します。**monFlags** プロパティで定義された遅延値に基づいて、重大度および現在の違反の条件を使用して評価されます。有効な値は以下のとおりです。

- unknown(1)
- ok(2)
- warning(3)
- minor(4)
- major(5)
- critical(6)
- fatal(7)
- up(11)
- down(12)

up および down の値は、レガシー モニタ（重大度が none）に対してのみ表示されます。この列の状態変更は、状態が集計状態テーブル内の対応する管理対象オブジェクトのすべてのモニタで最悪である場合にのみ、状態変更トランプを送信します。

権限：読み取り専用

サンプル セルフ モニタ テーブル エントリ

このセクションでは、ターゲット システムで 1 分のロード平均をモニタするように SystemEDGE エージェントに指定する、セルフ モニタ テーブル（見やすくするために区切られたテーブル）内のエントリの例を示します。

Index	Desc	Interval	Sample Type	OID	CurrVal
21	"1min load avg"	60	absolute Value	loadAvg 1Min.0	320
Oper	Value	Last Call	Num Traps	Last Trap	Row Status
gt	300	0d 22:40:00	0	0	active
Min Value	Max Value	Action	Flags	Superseded By	Obj Class
1	3	""	0x0	0	System
Obj Instance		Obj Attribute		Severity	Curr State
Load		LoadAverage1Min		warning	warning

このエントリは以下の情報を提供します。

- これはテーブル内の 21 番目の行です。
- その目的はシステムの 1 分のロード平均をモニタすることです。
- 60 秒ごとに、エージェントは、loadAvg1Min MIB 変数の絶対値（現在の値は 320）をチェックして、300（指定されたしきい値）を超えているかどうかを確認します。300 を超えている場合、エージェントは現在の状態を **warning**（重大度の値によって定義）に変更し、このモニタが集計状態テーブル内の対応する管理対象オブジェクトの最悪の状態を表す場合、状態変更トラップを送信します。
- 各ポーリングで、エージェントは、取得した値で CurrVal 列を更新し、LastCall 列の時間を記録し、必要に応じて（すなわち、ポーリングされた値がそれまでに観察された最低値または最高値である場合） MinValue 列と MaxValue 列を更新します。
- RowStatus 列は、このエントリがアクティブなことを示します。この例で、アクションは NULL であるため、トラップが送信されたときにコマンドは実行されません。

- **Flags** 列はゼロに設定されます。ゼロはデフォルトのセルフ モニタ テーブルの動作を示します。
- クラス、インスタンス、および属性値は、集計状態テーブル内でセルフ モニタ エントリをこれらの値の管理対象オブジェクトの一部として確立します。さまざまな重大度をモニタするために他のセルフ モニタ エントリをこれらの同じ値で定義することや、このエントリを他の関連オブジェクトで集計することができます。

行作成

セルフ モニタ テーブルの **monIndex** 列は、テーブル内の行を識別するためのキーフィールド(または行インデックス)として機能します。行 1 ~ 10 は、**SystemEDGE** エージェントによる内部使用に予約済みです。行は 11 ~ **MAXINT** の範囲で設定できます。

セルフ モニタ テーブルを備えた以下の MIB オブジェクトを使用して、行作成を最適化することができます。

monUnusedIndex

オブジェクト上で **SNMP Get** を実行したときに、セルフ モニタ テーブルの未使用のインデックス番号を返します。行作成に使用できる番号を取得するには、このオブジェクトをクエリします。

monMatchDescr

monDescr 属性内の特定のエントリの説明に対応するインデックス番号を特定します。 **monDescr** 属性値を指定してこの MIB オブジェクト上で **SNMP Set** を実行すると、エージェントがセルフ モニタ テーブル内のエントリを検索し、**monMatchIndex** MIB オブジェクト内で最後に一致したエントリのインデックス値を配置します。

monMatchIndex

monMatchDescr と共に使用したときに、特定のエントリの説明をそのインデックス番号と照合します。

ローカル ポリシーに基づいて、行のブロックをシステム管理者に予約するように選択することができます。その後、その行が別のユーザのエントリによってすでに取得されている場合があることを懸念せずに、行の予約済みブロック内のエントリを定義できます。その他のユーザがユーザ設定のエントリを定義する場合は、ローカル ポリシーに従って、予約された範囲外にある行インデックスを使用する必要があります。

行のブロックを予約することによって、一貫した条件のセット（行エントリ）がすべてのマシンにおいてモニタされるよう定義することが可能となり、各マシンの同じ行番号に同じ条件が定義されるようになります。たとえば、行 31

（monIndex = 31）を使用して swapCapacity 変数をモニタするためのエントリを定義することができます。また、この設定をすべてのシステムに配布できるため、行 31 はすべてのシステム上の swapCapacity 変数をモニタします。インデックス 11 ～ 16 は事前定義済みモニタに使用されます。リモート モニタリング AIM は、8000 を超えるインデックスを使用します。

CA Virtual Assurance でエージェントを設定すると、このプロセスを自動化できます。特定の行番号のエントリのブロックを設定し、1 回の操作ですべての管理対象システムにその設定を展開することができます。

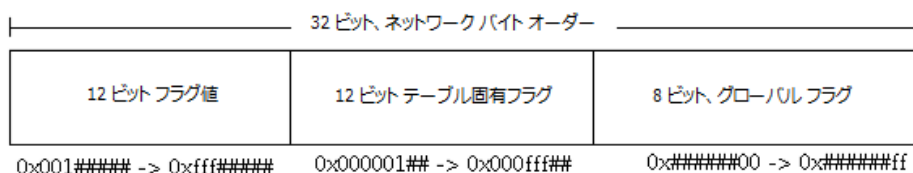
セルフ モニタ テーブルのフラグ

セルフ モニタ テーブル内の monFlags 列は 32 ビット符号なし整数で、対応するセルフ モニタ テーブル行の追加動作セマンティックを指定できます。デフォルトでは、セルフ モニタ テーブル行は以下を実行します。

- システム リソースのモニタ
- ステータスの評価
- SNMP トラップの送信（レガシー モニタのみ）
- sysedge.log ファイルへのトラップのログ記録
- アクションの呼び出し（アクションが設定されている場合）

これらのデフォルトを変更するようにフラグ ビットを設定できます。

SystemEDGE エージェントは、すべてのフラグを 16 進（ベース 16）法で解釈します。以下の図は、セルフ モニタ テーブルフラグ フィールド（monFlags）の構成を示します。



フラグ値は 3 つのフィールドから構成されます。

- フィールド 1 には、Systems Management Empire MIB のモニタ テーブル用に定義された共通のテーブル フラグが含まれます。この部分はフラグの低位 8 ビットです。
- フィールド 2 には、各モニタ テーブル用に個別に定義された、テーブル固有のフラグが含まれます。このフィールドは、共通テーブル フラグの次の低位 12 ビットを定義します。
- フィールド 3 には、テーブル固有のフラグで使用する整数値用の予約済み高位 12 ビットが含まれます。

以下のセクションでは、各フラグ ビットについて説明します。OR 論理演算によってこれらのフラグ値を組み合わせることができます。

以下のリストでは、セルフ モニタ テーブルのフラグについて説明します。

0x00000001

このエントリに対して、アクションが実行されないようにします。

0x00000002

このエントリに対して、SNMP トラップが送信されないようにします。

注: セルフ モニタ テーブルが制御するのは、レガシー モニタに対するトラップの送信のみです。ステートフルセルフ モニタについては、状態変更トラップが依然として集計状態テーブルから発生する場合があります。

0x00000004

このエントリを再初期化する試行を無効にします。デフォルトでは、エージェントはモニタしている MIB オブジェクトをクエリしようとすることによってこのエントリを定期的に再初期化しようとします。クエリが失敗した場合、再初期化が実行されます。

0x00000008

sysedge.log ファイル内で、このエントリに対し、トラップのログが記録されないようにします。このビットを設定しても、トラップの送信には影響しません。イベントが頻繁に発生する場合、または特定のエントリをエージェントのハートビートとして使用している場合は、イベントのログ記録を無効にすると有用です。

0x00000010

エージェントがモニタリングを再初期化しようとして MIB オブジェクトのクエリに失敗するたびに、このエントリに対して継続的な **monitorEntryNotReady** トラップを送信します。エージェントのデフォルトの動作は、エージェントがモニタしている MIB オブジェクトが存在しなくなったとき、その後のエントリの再初期化試行が失敗した場合でも、単一の **monitorEntryNotReady** トラップを送信することです。この機能を有効にすると、エージェントは、再初期化が失敗するたびに追加の **monitorEntryNotReady** トラップを送信します。

0x00000020

アクション スクリプトまたはプログラムへのデフォルト引数の受け渡しを無効にします。SystemEDGE は、通常、トラップタイプ、説明フィールドなどを示すデフォルト アクション パラメータを渡します。アクション パラメータの詳細については、「[セルフ モニタ テーブルのアクション パラメータ \(P. 249\)](#)」を参照してください。

0x00000040

このエントリに対して **notReady** トラップが送信されないようにします。**notReady** トラップのログは記録されず、**notReady** トラップに対してアクションも実行されません。

0x00000100

セルフ モニタ 表現が **True** から **False** に移行したとき、このエントリに対して **monitorClear** トラップを送信します。これには **monitorClear** トラップのアクションのログ記録と実行が含まれます。

0xXXX00200

モニタしきい値トラップを送信するか、または *X* 回目の連続する違反でのみ現在の状態を変更します。*X* 回目の違反が発生した後、エージェントは後続の **True** 表現評価ごとにモニタ トラップを送信します。しきい値表現が **True** から **False** に移行すると、エージェントはゼロから後続の違反を数え始めます。このフラグは、アクションの実行およびログ記録にも適用されます。フラグ値フィールドから *X* の値を指定できます。この動作の例については、「[セルフ モニタリングの例 \(P. 260\)](#)」を参照してください。

0xXXX00400

最大 X の連続するモニタ トラップを送信し、それ以上送信しません。この機能を有効にすると、しきい値を超えたときに発生する可能性のある連続するモニタ トラップおよびアクション実行の数の上限が設定されます。しきい値表現が **True** から **False** に移行すると、エージェントはゼロから後続の違反を数え始めます。このフラグは、アクションの実行およびログ記録にも適用されます。フラグ値フィールドから X の値を指定できます。この動作の例については、「[セルフ モニタリングの例 \(P. 260\)](#)」を参照してください。

0x00000800

オブジェクト情報（クラス、インスタンス、属性）および重大度が同じモニタの状態を集計するため、モニタには **AND** 関係があります。このフラグセットがある同じ重大度の接続されたすべてのモニタには、その都度変化する集計状態に対する違反条件が必要です。

0x###00000

いくつかのフラグ ビットでは、トラップを送信し、アクションとログを実行するために、値 X が使用されます。値 X はフラグ フィールドの高順位の 12 ビットとして指定されます。このフィールドを利用するフラグ ビットは相互に排他的です。

セルフ モニタ テーブルのアクション パラメータ

SystemEDGE エージェントには、呼び出されたアクション コマンドに対していくつかのデフォルト パラメータが用意されています。これらのパラメータは、アクション文字列で指定したパラメータに加えられ、指定したパラメータの後のコマンドラインで渡されます。デフォルト アクション パラメータは、エージェントがセルフ モニタ テーブルに対して送信する **SNMP** トラップ内に用意されているパラメータと同じです。

注: テーブル エントリで **0x00000020** フラグを設定することによって、エージェントがこれらのアクション パラメータをコマンドに含めないようにすることができます。

以下のリストでは、セルフ モニタ テーブルのアクションのデフォルト パラメータについて説明します。

trapType

ntEventMonNotReadyTrap や **ntEventMonMatchTrap** など、送信されるトラップのタイプを定義します。

monDescr

テーブル エントリの説明を定義します。

monOID

テーブル エントリの **OID** を定義します。

monCurrVal

テーブル エントリの現在の値を定義します。

monValue

テーブル エントリのモニタされている値を定義します。

monRowStatus

テーブル エントリの現在の行ステータスを定義します。

monOperator

テーブル エントリの使用されている演算子を定義します。アクション スクリプトに渡される値は実際の演算子の数値表現です。たとえば、演算子が **>** である場合、アクション スクリプトに渡される値は **2** です。

monIndex

テーブル エントリのインデックスを定義します。

monFlags

テーブル エントリ と関連付けられたフラグを定義します。アクション スクリプトに渡される値は、16 進数（たとえば **0x00000020** など）であることを示すために、先頭が **0x** の 16 進表記になっています。

monObjClass

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト クラスを定義します。エージェントは、**sysedge.oid** ファイルを使用してデフォルト値を割り当てます。クラス名はモニタされたオブジェクト テーブルを指定するために使用できます。**SystemEDGE** が指定された名前を持つ既存のテーブルを決定しない場合、**SystemEDGE** はクラスを作成します。クラスはモニタリング オブジェクト識別子 (**SNMP OID**) を指定し、集計目的で参照できます。

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

monObjInstance

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト インスタンスを定義します。値が設定されていない場合、エージェントでデフォルトを判断できなければ、値は「不明」になります。表形式以外のオブジェクトのデフォルトは「**NULL**」です。リモートオブジェクトのシームレスなモニタリングの場合、インスタンス名はプレフィックス **//hostname/** で開始されます。

ローカル オブジェクトの場合は、エージェントがプレフィックス **//.** を各インスタンス名に追加します（「**NULL**」を含む）。この属性は、空文字列または「*****」に設定することはできません。そのような要求は無視されます。インスタンス名が対応するインスタンスへの解析時に一致できない場合、**SystemEDGE** はこのモニタリング エントリの **OID** を指定されたインスタンス名で作成します。

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

monObjAttribute

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト インスタンスを定義します。エージェントは、**sysedge.oid** ファイルを使用して、**MIB** 属性名に対応するオブジェクトのデフォルト値を割り当てます。属性名がモニタリング エントリの解析時に不明な場合、**SystemEDGE** は指定された名前属性を作成します。

権限：読み書き

デフォルト：モニタされる **OID** の **ASCII MIB** 属性名

重要：新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

monCurrState

テーブル エントリの現在の状態を定義します。

セルフ モニタ テーブルの表示

CA Virtual Assurance を使用して、管理対象エージェントのセルフ モニタ テーブルの内容を表示することができます。

CA Virtual Assurance でセルフ モニタ テーブルを表示する方法

1. CA Virtual Assurance ユーザ インターフェースにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペイン内の [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバの詳細のページが表示されます。
4. [設定] をクリックし、[セルフ モニタ] タブをクリックします。

セルフ モニタ テーブルが、選択されたサーバ上のすべての既存のセルフ モニタと共に表示されます。

セルフ モニタリングの設定

セルフ モニタ テーブルでエントリを追加、削除、または変更することによって、SystemEDGE エージェントがモニタする MIB 変数と条件を制御することができます。

セルフ モニタ テーブルは以下の方法で設定できます。

CA Virtual Assurance からのファイルベース設定の動的使用

CA Virtual Assurance では、管理しているシステムにインストールされているエージェントを **CA Virtual Assurance** ユーザ インターフェースから管理できます。エージェントの再起動が必要ないユーザ フレンドリーで安全なメカニズムを備えたモニタを動的に定義して送信する機能が備わっているため、これは設定の推奨された方法です。個々のエージェントのモニタを設定するか、または複数の管理対象システムで適用するモニタ ポリシーを定義できます。個々の設定を変更するか、設定ポリシーを適用すると、**CA Virtual Assurance** によりエージェントシステムに新しい設定ファイルが送信されます。**CA Virtual Assurance** は、データ センター全体におけるエージェント設定の中心点の役割を果たします。また、他の方法によって加えられた変更を上書きしたり、拒否することができます。

CA Virtual Assurance を使用したエージェントの設定の詳細については、**CA Virtual Assurance** のドキュメントを参照してください。

eHealth AdvantEDGE View、CA Spectrum、または CA NSM などの管理システムからの SNMP Set コマンドの動的使用

さまざまな管理システムで、**SNMP Set** コマンドによる SystemEDGE へのモニタの追加がサポートされます。**eHealth** では、**AdvantEDGE View** インターフェースを介してこの機能を利用できます。また、**CA NSM** ではエージェント ビューを介して設定が可能で、**CA Spectrum** では **OneClick** を介して設定が可能です。**CA Virtual Assurance** を介した動的設定により利用可能な機能はどれも、これらの方法では利用できません。

SNMP Set の設定の詳細については、該当する管理システム用のドキュメントを参照してください。

ローカル sysedge.cf ファイルでのセルフ モニタリング テーブルのエントリの手動指定

マニュアル設定では、特定のディレクティブを使用してローカル **sysedge.cf** ファイルにエントリを追加する必要があります。 **sysedge.cf** ファイルからエージェントを直接設定した場合、変更を適用するには完全な再起動が必要です。 直接設定するにはエージェント データ ディレクトリにある **sysedge.cf** ファイルを使用します。

このセクションでは、ローカル **sysedge.cf** ファイルでセルフ モニタリングを直接設定する方法について説明しますが、あらゆる種類の設定に必要なプロパティについて説明します。

monitor ディレクティブ -- セルフ モニタ テーブルへのエントリの追加

monitor ディレクティブを使用すると、**sysedge.cf** ファイルでセルフ モニタ テーブルにモニタ エントリを直接追加できます。引数はセルフ モニタ テーブルの列を表します。

以下で説明する構文を使用してエージェント データ ディレクトリ内の **sysedge.cf** ファイルに行を追加し、ファイルを保存し、エージェントを再起動して変更を適用します。

monitor ディレクティブを使用して、セルフ モニタ テーブルにエントリを以下のように追加します。

```
(1) monitor oid oid index flags interval stype oper thresh ['descr' ['action' [supIndex]]]
```

```
(2) monitor oid oid index flags interval stype oper thresh ['descr' ['action' ['objClass' 'objInst' 'objAttr' severity]]]
```

```
(3) monitor table inst attr index flags interval stype oper thresh ['descr' ['action' [supIndex]]]
```

```
(4) monitor table inst attr index flags interval stype oper thresh ['descr' ['action' ['objClass' 'objInst' 'objAttr' severity]]]
```

oid

モニタリングする **OID** を指定します。ドットで区切った完全な **10** 進数値（たとえば、**1.3.6.1.2.1.25.1.5.0** など）またはシンボリック **MIB** 名（たとえば、**hrSystemNumUsers.0** など）を使用して、**OID** を指定できます。いずれかの方法で、オブジェクト インスタンスを指定する必要があります。オブジェクト インスタンスは、表形式以外の **MIB** 変数の場合は通常ゼロであるか、**MIB** テーブルに属する属性の特定のインスタンスに対応します。

例：

- 1.3.6.2.1.25.1.5.0
- hrSystemNumUsers.0

table 'instance' attribute

モニタするオブジェクトの **tableEntry**、インスタンス、およびモニタするオブジェクトの属性名を指定します。属性にインスタンスがない場合、インスタンスプロパティに対して " を入力します。

例：

- processGroupMonEntry 'iexplore|IEXPLORE' pgmonRSS
- devTableEntry 'C:' devCapacity

詳細については、「[OID の逆ルックアップ](#) (P. 258)」を参照してください。

index

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 1 ～ 10 はエージェントによる内部使用に予約済みです。したがって、インデックス値は 10 より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば 0x00000001）を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグの詳細については、「[セルフ モニタ テーブルのフラグ](#) (P. 245)」を参照してください。

interval

モニタを実行する必要がある頻度を秒単位で指定します。たとえば、値 30 は、このエントリを 30 秒ごとにモニタするようにエージェントに指示します。最小値は 30 で、30 秒の倍数で指定する必要があります。

stype

エージェントが属性の絶対値 (**absoluteValue**) をサンプリングすべきか、または連続するサンプルの差異 (**deltaValue**) をとるべきかを指定します。変化のレートを知るためにカウンタ変数をモニタするには、**deltaValue** を使用します。正確な値を知るためにゲージをモニタするには、**absoluteValue** を使用します。

oper

現在のエントリ値を提供されたしきい値と比較評価するためのブール演算子を指定します。

演算子として有効な値は、以下のとおりです。

- nop (演算なし)
- > (より大きい)
- < (より小さい)
- >= (以上)
- <= (以下)
- = (等しい)
- != (等しくない)

デフォルト: **nop**

thresh

指定された演算子を使用して、モニタ対象属性の現在値と比較される整数しきい値を指定します。

'descr'

テーブルエントリの任意の説明(0 ～ 512 文字の長さ)を指定します。

'action'

しきい値条件が満たされ、エージェントがしきい値違反トラップ(レガシー モニタ)または状態変更トラップ(ステートフル モニタ)を送信したときに実行されるコマンド(長さが 0 ～ 2048 文字)を、フルパスおよび任意のパラメータを含めて指定します。ステートフルセルフモニタについては、しきい値違反がオブジェクトの状態変更をもたらさない場合、コマンドは実行されません。文字列が空の場合、エージェントはこのエントリのアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定は変更できます。詳細については、「エージェントの設定」の章を参照してください。

supIndex

親として機能するテーブルインデックスまたはこのエントリの最も重要なパートナーを指定します。有効な値は、既存または将来の **SystemEDGE** モニタ インデックス(正の整数値)です。デフォルトはゼロ(0)で、デフォルトの動作を示します。

オブジェクト状態モデルの使用(none 以外の重大度を入力)は廃止されており、この機能は無効になっています。

'objClass'

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクトクラスを定義します。エージェントは、**sysedge.oid** ファイルを使用してデフォルト値を割り当てます。クラス名はモニタされたオブジェクトテーブルを指定するために使用できます。**SystemEDGE** が指定された名前を持つ既存のテーブルを決定しない場合、**SystemEDGE** はクラスを作成します。クラスはモニタリングオブジェクト識別子 (**SNMP OID**) を指定し、集計目的で参照できます。

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

'objInst'

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクトインスタンスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を (**objClass** および **objAttr** と共に) 定義します。エージェントは、**sysedge.oid** ファイルを使用して、特定の属性値に基づいて表形式オブジェクトのデフォルト値を割り当て、非表形式オブジェクトのデフォルトとして **NULL** を割り当てます。

'objAttr'

状態管理モデルで使用するためにオブジェクト属性を指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を (**objClass** および **objInst** と共に) 定義します。エージェントは、**sysedge.oid** ファイルを使用して、**MIB** 属性名に対応するデフォルト値を割り当てます。

デフォルト: MIB 属性名

重大度

オブジェクト状態管理モデルに使用する重大度を指定します。この値は、しきい値違反が発生したときにエントリに割り当てる状態を定義します。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

none の値はレガシー モニタを作成し、エントリがオブジェクト状態管理モデルから除外されます。

デフォルト : **none**

OID の逆ルックアップ

エージェントインストールの設定ディレクトリ内にある `sysedge.oid` では、サポートされているすべての SystemEDGE MIB 属性のクラス、インスタンス、および属性プロパティのデフォルト値が管理されます。このファイルでは、デフォルトのクラス、インスタンス、および属性値を割り当てるために、以下の規則が使用されます。

クラス

属性が属する MIB グループまたはテーブル名を使用します。

例 :

- kernelConfig
- userEntry

インスタンス

MIB テーブルに属さない属性に対して、値「**NULL**」を使用します。表形式オブジェクトの場合、エージェントは、特定の属性または **MIB** テーブルの属性の値に基づいて、デフォルト インスタンス値を取得します。

例：

- devMntPt 属性の値（デバイス テーブル）
- processName 属性の値（プロセス テーブル）

属性

MIB 属性名を使用します。

例：

- devDevice
- maxProcs

OID の逆ルックアップの例については、「[セルフ モニタリングの例](#) (P. 260)」を参照してください。

セルフモニタリングの例

このセクションでは、ターゲットシステム上でしきい値をモニタするための、セルフモニタテーブルのサンプルエントリを示します。それぞれの例では、エントリを定義する方法を示し、モニタされている条件について説明します。これらのエントリは `sysedge.cf` に追加できます。

例: 1 分のロード平均のモニタ

以下の例では、システムの 1 分のロード平均をモニタするようにエージェントを設定します。

```
monitor oid 1.3.6.1.4.1.546.1.1.7.8.26.0 11 0x00 60 absolute > 300 'Monitor 1 minute  
load average' '' 'kernelperf' 'null' 'loadAverage1Min' warning
```

1.3.6.1.4.1.546.1.1.7.8.26.0

Systems Management Empire MIB 内に含まれている
loadAverage1Minute 変数の **OID** に対応します。

11

このエントリがモニタテーブルの行 **11** (**monIndex=11**) を占めることを示します。

60

ロード平均が **60** 秒に **1** 回サンプリングされるように指定します。

絶対値

エージェントが、連続するサンプルの差異ではなく、オブジェクトの値を使用する必要があることを示します。

300

現在のロード平均が比較される値を示します。現在サンプリングされている値が **300** より大きい (**>**) 場合、イベントが発生します。

注: エージェントは、基盤となるシステムのロード平均に **100** を掛けた値としてロード平均を返します。たとえば、3 のロード平均が **300** として返されます。

kernelperf

このエントリでオブジェクト状態モデルの **kernelperf** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

null

このエントリでオブジェクト状態モデルの **null** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注:「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「/*/*」は無効な値です。

loadAverage1Min

このエントリでオブジェクト状態モデルの **loadAverage1Min** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

警告

このエントリでオブジェクト状態モデルの **warning** 重大度が使用されることを示します。

例: 5 分のロード平均のモニタ

以下の例では、システムの 5 分のロード平均をモニタするようにエージェントを設定します。

```
monitor oid loadAverage5Min.0 12 0x00500300 300 absolute > 200 'Monitor 5 minute load average' '' 'kernelperf' 'null' 'loadAverage5min' fatal
```

loadAverage5Min.0

Systems Management Empire MIBis 内に含まれている

loadAverage5Minute 変数の **OID** に対応し、エントリはモニタ テーブルの行 12 (**monIndex=12**) を占めます。

0x00500300

- しきい値表現が **True** から **False** に移行したときに **monitorClear** トラップを送信するようにエージェントを設定します。
- イベントの **X** 回目の連続発生時にのみトラップの送信を開始する（およびアクションが設定されている場合にはそのアクションを実行する）ようにエージェントを設定します。

X=5 を指定します。その結果、エージェントはイベントの 5 回目の発生時にトラップの送信を開始します。

300

ロード平均が **300** 秒に 1 回サンプリングされるように指定します。

絶対値

エージェントが、連続するサンプルの差異ではなく、オブジェクトの値を使用する必要があることを示します。

200

現在のロード平均が比較される値を示します。現在サンプリングされている値が **200** より大きい (>) 場合、エージェントは設定されているすべてのマネージャにトラップを送信します。

"

アクションが指定されていないことを示します。

kernelperf

このエントリでオブジェクト状態モデルの **kernelperf** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

null

このエントリでオブジェクト状態モデルの **null** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/」は、無効な値です。

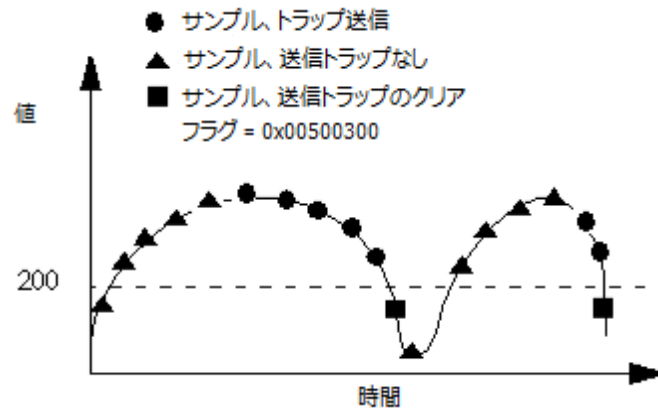
loadAverage5Min

このエントリでオブジェクト状態モデルの **loadAverage5Min** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

fatal

このエントリでオブジェクト状態モデルの **fatal** 重大度が使用されることを示します。

以下の図は、エージェントがこの **monitor** ディレクティブに基づいてトラップをどのように送信しクリアするかを示します。



例: 15 分のロード平均のモニタ

以下の例では、システムの 15 分のロード平均をモニタするようにエージェントを設定します。

```
monitor oid loadAverage15Min.0 13 0x0 900 absolute > 200 'Monitor 15 minute load
average' ' ' 'kernelperf' 'null' 'loadAverage15Min' critical
```

loadAverage15Min.0

Systems Management Empire MIB 内に含まれている **loadAverage15Min** 変数の **OID** に対応し、エントリはモニタ テーブルの行 **13** (**monIndex=13**) を占めます。

900

ロード平均が **900** 秒に **1** 回サンプリングされるように指定します。

絶対値

エージェントが、連続するサンプルの差異ではなく、オブジェクトの値を使用する必要があることを示します。

200

現在のロード平均が比較される値を示します。現在サンプリングされている値が **200** より大きい (**>**) 場合、エージェントは設定されているすべてのマネージャにトラップを送信します。

kernelperf

このエントリでオブジェクト状態モデルの **kernelperf** オブジェクト クラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

null

このエントリでオブジェクト状態モデルの **null** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注:「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/*」は、無効な値です。

loadAverage15Min

このエントリでオブジェクト状態モデルの **loadAverage15Min** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

critical

このエントリでオブジェクト状態モデルの **critical** 重大度が使用されることを示します。

例: システムの割り込み率のモニタ

以下の例では、ローカルシステム上でハードウェア割り込みが発生している割合をモニタするようにエージェントを設定します。

```
monitor oid numInterrupts.0 1003 0x00500400 60 delta > 1000 'Monitor Interrupt Rate'
'' 'kernelperf' 'null' 'numInterrupts' warning
```

numInterrupts.0

Systems Management Empire MIB 内に含まれている **numInterrupts** カウンタ オブジェクトの **OID** に対応し、エントリはモニタ テーブルのインデックス **14** です。

0x00500400

- このモニタ表現が **True** として評価されたときに最大 **X** の連続するトラップを送信するようにエージェントを設定します。
- **X=5** を指定します。その結果、エージェントは **5** つの連続する **monitorEvent** トラップを送信し、エントリがそれ自体をリセットするまでトラップの送信を停止します。表現が **True** から **False** に移行したとき、エントリはそれ自体リセットします。

エージェントが **monitorClear** トラップを送信するように指定しないでください。そのように指定すると、表現が **True** から **False** に移行したときに **monitorClear** トラップが送信されなくなります。

60

割り込み率が 60 秒ごとにサンプリングされることを示します。

差分

割り込みの数が増減した割合を測定するようにエージェントに指定します。このオブジェクトはカウンタであるため、デルタは適切なサンプルタイプです。

1000

現在の割り込み数が比較される値を示します。

kernelperf

このエントリでオブジェクト状態モデルの **kernelperf** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

null

このエントリでオブジェクト状態モデルの **null** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/」は無効な値です。

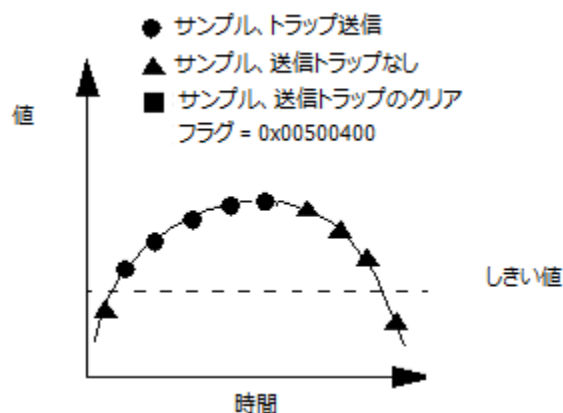
numInterrupts

このエントリでオブジェクト状態モデルの **numInterrupts** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

warning

このエントリでオブジェクト状態モデルの **warning** 重大度が使用されることを示します。

以下の図は、エージェントがこの **monitor** ディレクティブに基づいてトラップをどのように送信しクリアするかを示します。



例: システムのページフォールト率のモニタ

以下の例では、ローカルシステム上でハードウェア ページ割り込みが発生している割合をモニタするようにエージェントを設定します。

```
monitor oid numPageFaults.0 15 0x00500500 60 delta > 1000 'Monitor Page-fault Rate'
'' 'kernelperf' 'null' 'numPageFaults' major
```

numPageFaults.0

Systems Management Empire MIB 内に含まれている numPageFaults カウンタ オブジェクトの OID に対応し、エントリはモニタ テーブルのインデックス 15 です。

0x00500500

- 表現が **True** から **False** に移行したときに **monitorClear** トラップを送信するようにエージェントを設定します。
- エージェントが最大 **X** の連続する **monitorEvent** トラップを送信し、エントリがそれ自体をリセットするまでそれ以上送信しないように指定します。
- **X=5** を指定します。その結果、エージェントは **5** つの連続する **monitorEvent** トラップを送信し、エントリがそれ自体をリセットするまでトラップの送信を停止します。

60

割り込み率が **60** 秒ごとにサンプリングされることを示します。

差分

割り込みの数が増減した割合を測定するようにエージェントに指定します。このオブジェクトはカウンタであるため、デルタはこのエントリのサンプルタイプに対して適切な値です。

1000

現在の割り込み数が比較される値を示します。

kernelperf

このエントリでオブジェクト状態モデルの **kernelperf** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

null

このエントリでオブジェクト状態モデルの **null** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/*」は、無効な値です。

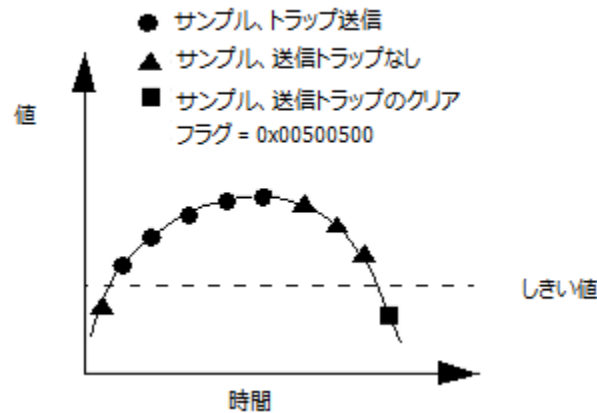
numPageFaults

このエントリでオブジェクト状態モデルの **numPageFaults** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

major

このエントリでオブジェクト状態モデルの **major** 重大度が使用されることを示します。

以下の図は、エージェントがこの **monitor** ディレクティブに基づいてトラップをどのように送信しクリアするかを示します。

**例：インターフェース上の着信パケット数のモニタ**

以下の例では、最初のイーサネットインターフェース（Sun のシステムの場合は **le0**）によってパケットが受信される割合をモニタするようにエージェントを設定します。

```
monitor oid ifInUcastPkts.2 1004 0x0 60 delta > 1000 'Monitor le0 Incoming Packets'
'' 'ifEntry' 'WAN MiniPort' 'ifInUcastPkts' minor
```

ifInUcastPkts.2

サンプリングする特定の **MIB** オブジェクトインスタンスを示します。

16

エントリがモニタ テーブルのインデックス **16** であることを示します。

60

エージェントが **60** 秒ごとに割合を計算することを示します。

差分

着信パケット (**ifInUcastPkts**) の数が増えている割合を測定するようにエージェントに指定します。

1000

比較で使用する値を指定します。割合の変化が 1000 より大きい (>) 場合、エージェントはトラップを送信します。

ifEntry

このエントリでオブジェクト状態モデルの **ifEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

WAN MiniPort

このエントリでオブジェクト状態モデルの **WAN MiniPort** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*instance」、および「//*/*」は、無効な値です。

ifInUcastPkts

このエントリでオブジェクト状態モデルの **ifInUcastPkts** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

minor

このエントリでオブジェクト状態モデルの **minor** 重大度が使用されることを示します。

例: インターフェース上の送信パケット数のモニタ

以下の例では、最初のイーサネットインターフェース（Sun のシステムの場合は **le0**）によってパケットが送信される割合をモニタするようにエージェントを設定します。

```
monitor oid ifOutUcastPkts.2 1005 0x0 60 delta > 1000 'Monitor le0 Outgoing Packets'  
' 'ifEntry' 'WAN MiniPort' 'ifOutUcastPkts ' ok
```

ifOutUcastPkts.2

サンプリングする特定の **MIB** オブジェクト インスタンスを示します。

60

エージェントが **60** 秒ごとに割合を計算することを示します。

delta

モニタされているオブジェクトが **MIB-II ifEntry** カウンタであるため、サンプルタイプを示します。

1000

比較で使用する値を指定します。割合の変化が **1000** より大きい (>) 場合、エージェントは設定されているすべてのマネージャにトラップメッセージを送信します。

ifEntry

このエントリでオブジェクト状態モデルの **ifEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

WAN MiniPort

このエントリでオブジェクト状態モデルの **WAN MiniPort** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/」は、無効な値です。

ifOutUcastPkts

このエントリでオブジェクト状態モデルの **ifOutUcastPkts** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

ok

このエントリでオブジェクト状態モデルの **ok** 重大度が使用されることを示します。

例: 受信される SNMP パケットの数のモニタ

以下の例では、エージェントが **SNMP** リクエストを受信する割合をモニタするようにエージェントを設定します。

```
monitor oid snmpInPkts.0 18 0x0 30 delta > 4000 'Monitor SNMP Packets' '' 'snmp' 'null'
'snmpInPkts' minor
```

snmpInPkts.0

サンプリングする **MIB II** オブジェクト インスタンスを示します。

18

エントリがモニタ テーブルのインデックス **18** であるように指定します。

30

エージェントが **30** 秒ごとに割合を計算することを示します。

差分

オブジェクトがカウンタであるため、サンプル タイプを示します。

4000

比較で使用する値を指定します。割合の変化が **4000** より大きい (>) 場合、エージェントは設定されているすべてのマネージャにトラップメッセージを送信します。

snmp

このエントリでオブジェクト状態モデルの **snmp** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

null

このエントリでオブジェクト状態モデルの **null** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注:「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*instance」、および「//*/*」は、無効な値です。

snmplnPkts

このエントリでオブジェクト状態モデルの **snmplnPkts** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

minor

このエントリでオブジェクト状態モデルの **minor** 重大度が使用されることを示します。

例: ルートファイルシステム上の領域のモニタ

以下の例では、ルート (/) ファイルシステムをモニタし、使用領域が 95% を超えたときにトラップメッセージを送信するようにエージェントを設定します。

```
monitor filesystem / devCapacity 19 0x0 120 absolute > 95 'Monitor / Filesystem' ''  
'devTableEntry' '/' 'devCapacity' minor
```

devCapacity

モニタする特定の MIB オブジェクトインスタンスを示します。この場合、オブジェクトインスタンスは **Systems Management Empire MIB devTable** の **devTableEntry.devCapacity** です。オブジェクトインスタンスは、ファイルシステムの名前に基づいて自動的に決定されるため、指定されません。

19

このエントリがモニタ テーブルの行 19 であることを示します。

120

エージェントが 120 秒ごとにサンプリングすることを示します。

absolute

エージェントが、ファイルシステムの空き領域を表す整数（カウンタではない）値をサンプリングしているため、適切なサンプルタイプを示します。

95

比較で使用する値を指定します。ファイルシステムの使用領域が 95% を超える (>) と、エージェントは設定されているすべてのマネージャにトラップメッセージを送信します。

devTableEntry

このエントリでオブジェクト状態モデルの **devTableEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

「/」

このエントリでオブジェクト状態モデルの「/」オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*instance」、および「/*/*」は無効な値です。

devCapacity

このエントリでオブジェクト状態モデルの **devCapacity** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

minor

このエントリでオブジェクト状態モデルの **minor** 重大度が使用されることを示します。

例: /usr ファイルシステム上の領域のモニタ

以下の例では、/usr ファイルシステムをモニタし、使用領域が 95% を超えたときにトラップメッセージを送信するようにエージェントを設定します。

```
monitor filesystem /usr devCapacity 20 0x00100500 120 absolute > 95 'Monitor /usr
Filesystem' '' 'devTableEntry' '/usr' 'devCapacity' fatal
```

devCapacity

モニタする特定の **MIB** オブジェクトインスタンスを示します。この場合、オブジェクトインスタンスは **Systems Management Empire MIB devTable** の **devTableEntry.devCapacity** です。オブジェクトインスタンスは、ファイルシステムの名前に基づいて自動的に決定されるため、指定されません。

20

このエントリがモニタ テーブルの行 **20**であることを示します。

0x00100500

- 表現が **True** から **False** に移行したときに **monitorClear** トラップを送信するようにエージェントを設定します。
- エージェントが最大 **X** の連続する **monitorEvent** トラップを送信し、エントリがそれ自体をリセットするまでそれ以上送信しないように指定します。
- **X=1** を指定します。これは、エージェントが、エントリがそれ自体をリセットするのを待機する前に、**monitorEvent** トラップを 1 つだけ送信することを示します。

120

エージェントが **120** 秒ごとにサンプリングすることを示します。

absolute

エージェントが、ファイルシステムの空き領域を表す整数値をサンプリングしているため、適切なサンプルタイプを示します。

95

比較で使用する値を指定します。ファイルシステムの使用領域が **95%** を超える (>) と、エージェントは設定されているすべてのマネージャにトラップメッセージを送信します。

devTableEntry

このエントリでオブジェクト状態モデルの **devTableEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

/usr

このエントリでオブジェクト状態モデルの **/usr** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/」は無効な値です。

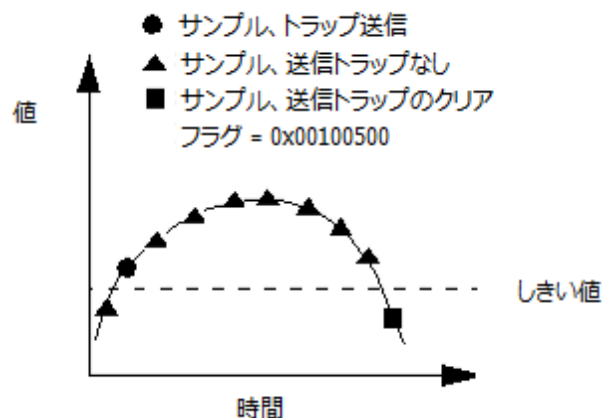
devCapacity

このエントリでオブジェクト状態モデルの **devCapacity** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

fatal

このエントリでオブジェクト状態モデルの **fatal** 重大度が使用されることを示します。

以下の図は、エージェントがこの **monitor** ディレクティブに基づいてトラップをどのように送信しクリアするかを示します。

**例: プロセス数のモニタ**

以下の例では、システム上で現在実行されているプロセス数をモニタし、その数が 120 より大きいときにトラップを送信するようにエージェントを設定します。

```
monitor oid hrSystemProcesses.0 21 0x0 60 absolute >= 120 'Monitor Number of Processes'
'' 'hrSystem' 'null' 'hrSystemProcesses' warning
```

hrSystemProcesses

モニタされる変数を示します。

21

このエントリがモニタ テーブルのインデックス **21** であることを示します。

60

エージェントが **60** 秒ごとにプロセス数をサンプリングすることを示します。

absolute

オブジェクトがゲージであるため、サンプルタイプを示します。

>=

プロセス数が 120 以上である場合は常に、トラップを送信するようにエージェントに指定します。

hrSystem

このエントリでオブジェクト状態モデルの **hrSystem** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

null

このエントリでオブジェクト状態モデルの **null** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/*」は、無効な値です。

hrSystemProcesses

このエントリでオブジェクト状態モデルの **hrSystemProcesses** オブジェクト属性が使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

warning

このエントリでオブジェクト状態モデルの **warning** 重大度が使用されることを示します。

セルフ モニタリング用の edgemon コマンド

edgemon コマンドおよび関連する引数は以下のとおりです。

```
oid objectOID index flags interval sampleType oper value "descr" "action"  
[supersededBy | objClass objInst objAttr severity]
```

```
filesystem|interface|cpu|disk name variable-name index flags interval sampleType  
oper value "descr" "action" [supersededBy | objClass objInst objAttr severity]
```

```
setStatus index status
```

```
delete index
```

```
list
```

|によって区切られた引数のリストは、左側のリストまたは右側のリストを使用できることを意味します。

objectOID

モニタリングする **OID** を指定します。**OID** は、完全なドット付き **10 進** 数値（たとえば **1.3.6.1.2.1.25.1.5.0**）またはシンボリック **MIB** 名（たとえば **hrSystemNumUsers.0**）を使用して指定できます。いずれかの方法で、オブジェクトインスタンスを指定する必要があります。オブジェクトインスタンスは、表形式以外の **MIB** 変数の場合は通常ゼロであり、**MIB** テーブルに属する属性の特定のインスタンスに対応します。

例：

- 1.3.6.2.1.25.1.5.0
- hrSystemNumUsers.0

index

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 **1 ~ 10** はエージェントによる内部使用に予約済みです。したがって、インデックス値は **10** より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば **0x00000001**）を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグの詳細については、「[セルフ モニタ テーブルのフラグ \(P. 245\)](#)」を参照してください。

interval

モニタを実行する必要がある頻度を秒単位で指定します。たとえば、値 30 は、このエントリを 30 秒ごとにモニタするようにエージェントに指示します。最小値は 30 で、30 秒の倍数で指定する必要があります。

sampleType

絶対値またはデルタのいずれかのサンプルタイプを指定します。この値は、エージェントがオブジェクトの絶対値をサンプリングするか、連続するサンプル間の差異を取得するかを示します。

oper

現在のエントリ値を指定されたしきい値と突き合わせて評価するためのブール演算子を指定します。

演算子の有効な値は以下のとおりです。

- nop (演算なし)
- > (より大きい)
- < (より小さい)
- >= (以上)
- <= (以下)
- = (等しい)
- != (等しくない)

デフォルト: **nop**

値

モニタされる属性の現在値が指定された演算子を使用して比較される整数しきい値を指定します。

"descr"

テーブルエントリの任意の説明 (長さが 0 ～ 512 文字) を指定します。

"action"

しきい値条件が満たされ、エージェントがしきい値違反トラップ（レガシー モニタ）または状態変更トラップ（ステートフル モニタ）を送信したときに実行されるコマンド（長さが 0 ～ 2048 文字）を、フルパスおよび任意のパラメータを含めて指定します。ステートフルセルフモニタについては、しきい値違反がオブジェクトの状態変更をもたらさない場合、コマンドは実行されません。文字列が空の場合、エージェントはこのエントリに対してアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定は変更できます。詳細については、「エージェントの設定」の章を参照してください。

supersededBy

親として機能するテーブル インデックスまたはこのエントリの最も重要なパートナーを指定します。有効な値は、既存または将来の **SystemEDGE** モニタ インデックス（正の整数値）です。デフォルトはゼロ（0）で、デフォルトの動作を示します。

オブジェクト状態モデルの使用（none 以外の重大度を入力）は廃止されており、この機能は無効になっています。

objClass

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクト クラスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を（**objInst** および **objAttr** と共に）定義します。エージェントは、**sysedge.oid** ファイルを使用してデフォルト値を割り当てます。

デフォルト: **MIB** グループまたはテーブル名（「**sysedge.oid**」を参照）

objInst

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクト インスタンスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を（**objClass** および **objAttr** と共に）定義します。エージェントは、**sysedge.oid** ファイルを使用して、特定の属性値に基づいて表形式オブジェクトのデフォルト値を割り当て、非表形式オブジェクトのデフォルトとして **NULL** を割り当てます。

objAttr

状態管理モデルで使用するためにオブジェクト属性を指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を (*objClass* および *objInst* と共に) 定義します。エージェントは、*sysedge.oid* ファイルを使用して、MIB 属性名に対応するデフォルト値を割り当てます。

デフォルト: MIB 属性名

severity

オブジェクト状態管理モデルに使用する重大度を指定します。この値は、しきい値違反が発生したときにエントリに割り当てる状態を定義します。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

none の値はレガシー モニタを作成し、エントリがオブジェクト状態管理モデルから除外されます。

デフォルト: なし

name

入力したコマンドのタイプ (*filesystem*、*interface*、*cpu*、または *disk*) に応じて、ファイルシステム、ネットワーク インターフェース、CPU、またはモニタするディスクの名前を指定します。たとえば、ファイルシステムの場合は *C:* です。

variable-name

入力したコマンド (*filesystem*、*interface*、*cpu*、または *disk*) に対応する、モニタする Systems Management Empire MIB テーブルの変数を指定します。たとえば、*variable-name* として *devCapacity* を指定することにより、ファイルシステムの容量をモニタできます。

status

エントリのステータスを定義します。エントリのステータスは以下のいずれかになります。

- active (行をアクティブ化する)
- notInService (行を非アクティブ化するが、保持する)
- destroy (セルフ モニタ テーブル行を削除する)

edgemon の例

このセクションでは、SNMP バージョン 1、2c、および 3 で edgemon コマンドを使用する方法の例を示します。

例: edgemon を使用した 1 分のロード平均のモニタ

以下の例では、しきい値が 3 の場合にシステムの 1 分のロード平均をモニタするエージェントのセルフ モニタ テーブルのインデックス 11 で、エントリを作成します。以下の例では IPv4 アドレスを示しますが、同様に IPv6 アドレスを指定できます。

```
edgemon -h fe80::901:dc19 -c private -v 1 -o oid loadAverage1Min.0 11 0x00 60 absolute  
'>' 300 "Monitor 1 minute load average" "" "" "" "" fatal
```

```
edgemon -h fe80::901:dc19 -c private -v 2c -o oid loadAverage1Min.0 11 0x00 60 absolute  
'>' 300 "Monitor 1 minute load average" "" "" "" "" minor
```

```
edgemon -h 143.45.0.12 -v 3 -u userName -s 3 -a authPassword -A MD5 -x encryptPassword  
-X DES -o oid loadAverage1Min.0 11 0x00 60 absolute '>' 300 "Monitor 1 minute load  
average" ""
```

例: edgemon を使用したハードウェア割り込みのモニタ

以下の例では、基盤となるシステムでのしきい値 1000 に対するハードウェア割り込みの数をモニタする、セルフ モニタ テーブル エントリを作成します。以下の例では IPv4 アドレスを示しますが、同様に IPv6 アドレスを指定できます。

```
edgemon -h fe80::901:dc19 -c private -v 1 -o oid numInterrupts.0 14 0x00500400 60 delta  
'>' 1000 "Monitor Interrupt Rate" ""
```

```
edgemon -h fe80::901:dc19 -c private -v 2c -o oid numInterrupts.0 14 0x00500400 60  
delta '>' 1000 "Monitor Interrupt Rate" "" "" "" "" minor
```

```
edgemon -h 143.45.0.12 -v 3 -u userName -s 3 -a authPassword -A MD5 -x encryptPassword  
-X DES -o oid numInterrupts.0 14 0x00500400 60 delta '>' 1000 "Monitor Interrupt Rate"  
""
```

エージェントはインデックス 14 でこのエントリを作成し、60 秒ごとに numInterrupts 変数をサンプリングします。0x00500400 のフラグ フィールドは、デフォルトのセルフ モニタ テーブルの動作を以下のように変更するようにエージェントに指定します。

0x00000400

最大 X の連続する monitorEvent トラップを送信し、それ以上送信しないようにエージェントに指定します。

0x00500000

ディレクティブで使用するために、フラグ値 $X=5$ が含まれます。

例: edgemon を使用した /usr ファイル システムのモニタ

以下の例では、2 分 (120 秒) ごとにファイル システムをチェックして、容量が 95% 以上になっていないかシステムの /usr ファイル システムをモニタするセルフ モニタ テーブル エントリを、インデックス 20 で作成します。以下の例では IPv4 アドレスを示しますが、同様に IPv6 アドレスを指定できます。

```
edgemon -h 143.45.0.12 -c private -v 1 -o filesystem /usr devCapacity 20 0x00100500
120 absolute '>=' 95 "Monitor /usr Filesystem" "" "" "" "" critical
```

```
edgemon -h 143.45.0.12 -c private -v 2c -o filesystem /usr devCapacity 20 0x00100500
120 absolute '>=' 95 "Monitor /usr Filesystem" ""
```

```
edgemon -h fe80::901:dc19 -v 3 -u userName -s 3 -a authPassword -A MD5 -x
encryptPassword -X DES -o filesystem /usr devCapacity 20 0x00100500 120 absolute '>='
95 "Monitor /usr Filesystem" "" "" "" "" critical
```

セルフ モニタリング エントリの削除

MIB 変数のモニタリングを停止するには、セルフ モニタ テーブルおよび sysedge.cf ファイルから該当するエントリを削除する必要があります。CA Virtual Assurance ユーザ インターフェースからポイント設定を使用してエントリを削除すると、両方の場所からエントリを削除できます。手動の削除では、個別の操作で両方の場所からエントリを削除する必要があります。

CA Virtual Assurance 内でセルフ モニタ エントリを削除する方法

1. CA Virtual Assurance ユーザ インターフェースにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペイン内の [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバの詳細のページが表示されます。
4. [設定] をクリックし、[セルフ モニタ] をクリックします。
セルフ モニタ テーブルが、すべての既存のセルフ モニタと共に表示されます。
5. 削除するモニタを選択し、[アクション] - [削除] をクリックします。
テーブルからエントリが削除されます。CA Virtual Assurance は、変更された `sysedge.cf` ファイルをエージェントにロードします。エージェントは、変更を適用するためにウォーム スタートを実行します。

セルフ モニタ エントリを手動で削除する方法

1. `-o` パラメータの以下のいずれかのコマンドを使用して、`edgemon` ユーティリティを実行します。

```
edgemon -o delete index
```

```
edgemon -o setstatus index 6
```

```
index
```

削除するセルフ モニタ エントリのインデックス値を指定します。

`setstatus` コマンドを使用するときに値を 6 に指定すると、行ステータスが破棄されるように設定され、エントリが削除されます。

注: `sysedge.cf` を編集した場合は、エージェントを再起動する必要があります。

2. ローカルの `sysedge.cf` ファイルからエントリの `monitor` ディレクティブを削除し、ファイルを保存します。
3. エージェントを再起動します。

第 9 章：プロセスとサービスのモニタリング

この章では、SystemEDGE エージェントを使用してプロセスとサービスをモニタする方法について説明します。エージェントは、プロセスが実行されているかどうか、プロセスのサイズ、CPU とメモリと使用率、ディスクとネットワーク I/O 操作の数など、プロセスの多くの属性をモニタできます。

このセクションには、以下のトピックが含まれています。

[プロセスとサービスのモニタリングの概要](#) (P. 285)

[Windows サービスのモニタリング](#) (P. 286)

[プロセス モニタの状態管理](#) (P. 287)

[プロセス モニタ テーブル](#) (P. 290)

[プロセス モニタリングの設定](#) (P. 315)

[プロセス モニタリング用の edgewatch コマンド](#) (P. 333)

[プロセス モニタリング エントリの削除](#) (P. 340)

[Solaris ゾーン プロセスのモニタリング](#) (P. 341)

[プロセスとサービスのモニタリングの推奨事項](#) (P. 342)

プロセスとサービスのモニタリングの概要

SystemEDGE エージェントでは、基盤となるシステム上で実行されている重要なプロセス、サービス、およびアプリケーションの特定の属性を動的にモニタできます。プロセス モニタ テーブルを使用して、プロセス、属性、しきい値、間隔、重大度、オブジェクト情報、およびその他の値を指定します。SystemEDGE エージェントは、定義されたプロセスを自動的にモニタし、指定した条件が満たされる場合には管理システムにトラップを送信します。

SystemEDGE エージェントは、指定されたあらゆるプロセス モニタの状態値を管理および更新し、オブジェクト クラス、インスタンス、および属性値を使用して、複数の定義済みモニタでオブジェクトのステータスを集計することができます。

エージェントはまた、管理対象システム上で管理機能を果たすコマンドを実行して、しきい値違反の原因をすぐに修正します。

プロセス グループのモニタリングの詳細については、「プロセス グループ モニタリング」の章を参照してください。

Windows サービスのモニタリング

UNIX システムでは、サービスやデーモンはプロセスで、他のプロセスと同様にモニタできます。それに対して、**Windows** システムでは、サービスはグラフィカル インターフェースを使用して（たとえば [コントロール パネル] の [サービス] から）開始および停止する特別な種類のプロセスです。**Windows** サービスはプロセス内で実行されますが、それらの間のマッピングは必ずしも 1 対 1 であるとは限りません。たとえば、複数の **Windows** サービスが単一のプロセス内で実行される場合があります。従って、以下の 2 つの方法で **Windows** サービスをモニタできます。

- **Windows** サービスの基盤となるプロセスをモニタすることによって、**Windows** サービスをモニタします。サービスが実行されている基盤となるプロセスを見つけるのが難しい場合があります。
- フラグを設定するか、設定キーワード ウォッチ `ntservice` を使用して、基盤となるプロセスの代わりに **Windows** サービスをモニタするように **SystemEDGE** エージェントに指定します。詳細については、「[プロセス モニタ テーブルのフラグ](#) (P. 305)」を参照してください。

注: **Windows** では **Windows** サービスのプロセス属性が追跡されないため、**SystemEDGE** エージェントは **Windows** サービスの `procAlive` 属性のみをモニタできます。特定の **Windows** サービスの他の属性（たとえば `procRSS`）をモニタするには、その **Windows** サービスを実装する基盤となるプロセスをモニタする必要があります。

プロセス モニタの状態管理

SystemEDGE 状態管理モデルは、プロセス モニタ テーブル内のすべてのエントリの状態値を管理し、ユーザがオブジェクト全体の状態を取得するために管理対象オブジェクトを定義できるようにします。状態管理モデルは、プロセス モニタ テーブル内のエントリに対して以下の機能を提供します。

- エージェントが各エントリの状態値を管理するために使用する、各プロセス モニタ エントリの重大度を定義する機能。重大度は CIM に準拠した状態に変換されます。
- オブジェクトクラス、インスタンス、および属性値を使用して、3 段階ネーミング スキームによって 1 つ以上の管理対象オブジェクトに 1 つ以上のモニタされたリソースをグループ化する機能。このメカニズムは柔軟であるため、ユーザはさまざまな機能領域のリソースを動的にグループ化できます。
- 特定のオブジェクトの状態が階層のより高いレベルのオブジェクトにプロパゲートされる、スーパー管理対象オブジェクトの階層を作成する機能。
- 管理対象オブジェクト全体の状態が変化した場合の状態変更トラップによる通知

状態管理モデルは、CA Virtual Assurance オブジェクト モデルに統合されます。AOM データベース内の管理対象オブジェクトはそれぞれ CIM に準拠した状態になり、SystemEDGE 管理対象オブジェクトおよび状態は SystemEDGE プラットフォーム管理モジュールによって CA Virtual Assurance に反映されます。CA Virtual Assurance によって SystemEDGE を管理すると、CA Virtual Assurance ユーザ インターフェース内の管理対象オブジェクトの状態情報を視覚化し、CA Virtual Assurance Event ビューを通して状態変更トラップを確認することができます。

CA Virtual Assurance 内での SystemEDGE 管理対象オブジェクトの視覚化の詳細については、CA Virtual Assurance のドキュメントを参照してください。

状態管理モデルを利用するためのプロセス モニタの設定の詳細（属性値および構文例を含む）については、この章の残りで説明します。

管理対象オブジェクトの作成

オブジェクトクラス、インスタンス、および属性値を使用して、プロセス モニタ テーブルから管理対象オブジェクトの作成を制御します。 デフォルトでは、**SystemEDGE** は、各一意のクラス、インスタンス、および属性プロセス モニタの管理対象オブジェクトを作成します。複数のプロセス モニタでこれらの 3 つのプロパティに対して同じ値が含まれる場合、エージェントは単一の管理対象オブジェクトにそれらをグループ化し、オブジェクト全体の状態を取得します。

ユーザがどのようにこのメカニズムを使用できるかの柔軟性を例証するには、以下のシナリオを検討します。

- プロセスの実メモリ使用率をモニタするために、クラス、インスタンス、および属性値が同じ 3 つのプロセス モニタ エントリを定義します。 モニタは、85% を超える実メモリ使用率に対して **warning** 重大度、90% を超える実メモリ使用率に対して **major** 重大度、および 95% を超える実メモリ使用率に対して **critical** 重大度を定義します。 エージェントは、これらのモニタを 1 つの管理対象オブジェクトに組み合わせ、現在の最悪の重大度をオブジェクト状態として管理します。
- ユーザの企業で接続されるさまざまな機能領域からプロセスをモニタするために、クラス、インスタンス、および属性値が同じ 3 つのプロセス モニタ エントリを定義します。たとえば、基本オペレーティング システム プロセス モニタ、アプリケーション プロセス モニタ、およびデータベース プロセス モニタを組み合わせ、各プロセス モニタに **warning** 重大度を割り当てることができます。 このシナリオでは、3 つのプロセス モニタしきい値すべてに違反する場合のみ、エージェントは警告状態変更トラップを送信します。
- オブジェクト インスタンスまたは属性に対するオブジェクト クラスとアスタリスク値が同じ値のプロセス モニタ エントリのスーパー集計を定義します。 このシナリオは、特にそれぞれの集計ルールを定義する必要なしに、多数のオブジェクトをグループ化します。たとえば、同じプロセスのさまざまな属性をモニタするすべてのオブジェクトを集計できます。その結果、そのプロセスのすべてのプロセス モニタは現在の最悪の状態をトップ レベル プロセス オブジェクトにプロパゲートします。

注: より高いレベルのすべてのオブジェクトを集計するようにエージェントを設定することもできます 詳細については、「[エージェントの設定](#)」の章を参照してください。

これらのシナリオの構文例については、「[プロセス モニタリングの例](#) (P. 324)」を参照してください。

ユーザがオブジェクト クラス、インスタンス、および属性値を定義しない場合、**SystemEDGE** は以下の規則に従ってこれらのプロパティのデフォルト値を割り当てます。

- クラス : `processEntry`

- インスタンス：モニタするプロセスを決定する正規表現を定義するプロセスモニタの `pmonRegExpr` 属性の値
- 属性：モニタするプロセス属性を定義するプロセスモニタの `pmonAttribute` 属性の値

関連項目

[オブジェクトステータス](#) (P. 71)

[オブジェクト集計](#) (P. 72)

[スーパー集計](#) (P. 74)

[状態管理と集計の仕組み](#) (P. 76)

集計状態テーブル

集計状態テーブルは、管理対象オブジェクト表現を作成し、オブジェクト全体の状態を取得するために必要な機能を果たします。集計状態テーブルは、**SystemEDGE** セルフモニタおよびプロセスモニタのフィルタされたビューです。テーブルには、管理対象オブジェクトの名前付けおよび各モニタエントリによって指定された状態情報に基づいて計算された、集計状態情報が含まれます。

テーブルは、オブジェクトクラス、インスタンス、および属性値（またはより高いレベルでグループ化されたオブジェクト）が同じすべてのセルフモニタおよびプロセスモニタの集計状態テーブルエントリを作成します。セルフモニタとプロセスモニタは、セルフモニタテーブルおよびプロセスモニタテーブル内に個別のエントリとしてのみ存在します。集計状態テーブルは、これらのモニタのグループ化を管理し、テーブルを連続的にポーリングし、管理対象オブジェクトの状態が変化したときに状態変更トラップを送信します。

注：新しい管理対象オブジェクトが作成されるか、既存の管理対象オブジェクトが削除されたときも、集計状態テーブルはトラップを送信します。

集計状態テーブルがどのように管理対象オブジェクトとそれらの状態を管理するかの詳細については、「コンセプト」の章を参照してください。テーブル列の説明については、「集計状態テーブル」を参照してください。

プロセス モニタ テーブル

プロセス モニタ テーブルは、エージェントが現在モニタしているプロセス（または Windows サービス）のそれぞれについての情報を提供します。行はそれぞれ、定義されたプロセス モニタ エントリを表します。

エントリごとに、テーブルは以下の種類の情報を提供します。

- インデックス
- 説明
- エージェントが変数をポーリングする間隔
- サンプル タイプ（絶対値またはデルタ値）
- モニタされている属性
- エージェントがモニタしている現在の値
- しきい値
- 最後の試呼
- すでに送信されたトラップの数
- 最後のトラップ
- フラグ
- トラップが送信されるときに講ずるべき処置
- 正規表現
- サンプリングされた最小値と最大値
- 現在のプロセス ID
- 行ステータス
- イベントの数
- オブジェクト クラス
- インスタンス
- 属性
- 定義された重大度
- 現在のモニタ ステータス
- ゾーン正規表現
- プロセス名

プロセス モニタ テーブルの列

以下のリストでは、プロセス モニタ テーブルの列について説明します。プロセス モニタ テーブルの詳細については、Systems Management Empire MIB の仕様（エージェントインストールの mib サブディレクトリ内の empire.asn1）を参照してください。

pmonIndex

このエントリの一意の行インデックス（1 ～ MAXINT）を定義します。

権限：読み取り専用

pmonDescr

モニタされているプロセスおよび属性の説明（長さが 0 ～ 512 文字）を定義します。

権限：読み書き

pmonInterval

数秒間にエージェントがプロセスをどれくらいの頻度でモニタするかを定義します。値は 30 秒以上かつ 30 秒の倍数にする必要があります。

権限：読み書き

デフォルト： 60

pmonSampleType

エージェントが属性の絶対値（absoluteValue(1)）をサンプリングするか、連続するサンプル（deltaValue(2)）間の差異を取得するかを定義します。たとえば、カウンタ属性をモニタするには deltaValue を使用します。deltaValue は変化率を示すからです。ゲージをモニタするには absoluteValue を使用します。absoluteValue はオブジェクトの正確な値を示すからです。

権限：読み書き

デフォルト： absoluteValue(1)

pmonAttribute

モニタされているプロセス属性を指定します。サポートされているすべての属性のリストについては、「[プロセス属性 \(P. 301\)](#)」を参照してください。たとえば、プロセスがアライブであることを確認するためにそのプロセスをモニタするには、procAlive 属性を指定します。特定のアプリケーションまたはプロセスによって受信されたパケットの数を追跡するには、procMsgsSent を指定します。

権限：読み書き

pmonCurrVal

モニタされているプロセスの最後に記録された属性値を指定します。
pmonInterval 秒ごとに、エージェントは属性の最新の読み取りを反映するためにこのフィールドを更新します。

procAlive をモニタするとき、この値は **hrSWRunStatus** 変数 (**hrSWRunTable** 内) または Windows サービスの場合は **ntServiceState** (**ntServiceTable** 内) からマッピングされます。**hrSWRunStatus** の有効な値は以下のとおりです。

1、2、または 3

プロセスが実行されていることを示します。

4

プロセスが実行されていないことを示します (この場合、そのプロセスは通常ゾンビ プロセスです)。

ntServiceState の有効な値は以下のとおりです。

1

サービスが実行されていることを示します。

2,3,4

サービスが実行されていないことを示します。

権限：読み取り専用

pmonOperator

表現 *currval* 演算子の値を評価するために使用されるブール演算子を指定します。演算子は以下のいずれかになります。

- **nop(1)** (演算なし。オブジェクトの値をモニタするが、ブール式を評価しない)
- **gt(2)** (次の値より大きい)
- **lt(3)** (次の値より小さい)
- **ge(4)** (次の値以上)
- **le(5)** (次の値以下)
- **eq(6)** (等しい)
- **ne(7)** (等しくない)

権限：読み書き

デフォルト： **nop(1)**

pmonValue

各モニタリング サイクル中にモニタされるプロセス属性の現在の値をエージェントが比較する整数値を定義します。比較が **True** として評価された場合、エージェントはモニタ状態を変更するか（重大度が **none** 以外のプロセス モニタの場合）、しきい値違反トラップを送信します（重大度が **none** のプロセス モニタの場合）。たとえば、プロセスの実メモリ使用率が **75** パーセントを超える場合に通知されるようにする場合は、エージェントが属性の現在の値を比較する **pmonValue** として **75** を設定します。

権限：読み書き

pmonLastCall

エージェントがモニタしているプロセス属性を最後にサンプリングした（呼び出した）時間を（**sysUpTime** に基づいて）定義します。**0** は、プロセスがまだサンプリングされていないことを示します。

権限：読み取り専用

pmonNumTraps

このエントリに送信されたトラップの数を定義します。この値は **processStopTrap**、**processStartTrap**、**processClearTrap**、**processNotReadyTrap**、および **processThresholdTrap** を追跡します。これらは、重大度が **none** のプロセス モニタでプロセスが停止したとき、またはしきい値違反が発生したときにエージェントによって送信されます。この値は、レガシー モニタに対して例外条件が発生している頻度を決定するのに役立ちます。この値は、見つからないトラップ メッセージを検出する手段も提供します。

権限：読み取り専用

注：ステートフル プロセス モニタの管理対象オブジェクトに対して送信された状態変更トラップの数の詳細については、「集計状態テーブル」を参照してください。

pmonLastTrap

エージェントがこのエントリに最後にトラップを送信した時間を (**sysUpTime** に基づいて) 定義します。0 は、トラップが送信されていないことを示します。この値は **processStopTrap**、**processStartTrap**、**processClearTrap**、**processNotReadyTrap**、および **processThresholdTrap** を追跡します。これらは、重大度が **none** のプロセス モニタでプロセスが停止したとき、またはしきい値違反が発生したときにエージェントによって送信されます。

権限：読み取り専用

注: ステートフル プロセス モニタの管理対象オブジェクトに対して送信された最後の状態変更トラップの詳細については、「集計状態テーブル」を参照してください。

pmonFlags

エントリがその操作の過程で従う追加の動作セマンティクスを示す正の整数フラグを定義します。利用可能なフラグおよびフラグ設定の詳細については、「[プロセス モニタ テーブルのフラグ \(P. 305\)](#)」を参照してください。

権限：読み書き

pmonAction

レガシー トラップ (レガシー モニタ) または状態変更トラップ (ステートフル モニタ) を送信したときに実行される引用符で囲まれたコマンド (長さが 0 ~ 2048 文字) を、任意のパラメータと共に定義します。ステートフル プロセス モニタについては、しきい値違反がオブジェクトの状態変更をもたらさない場合、コマンドは実行されません。文字列が空の場合、エージェントはこのエントリのアクションを実行しません。

注: アクションに **Windows** バッチ ファイルを使用しないでください。プログラム上の厳しい制限があり、デスクトップ アプリケーションでは正しく動作しないことがよくあります。代わりに、**Perl** または **Visual Basic** など、強力な柔軟なスクリプト言語を使用してください。

権限：読み書き

pmonRegExpr

モニタするアプリケーションまたはプロセスのプロセス ID をエージェントが取得しようとするときに適用される正規表現を指定します。**Windows** サービス モニタリングの場合、この正規表現はモニタする **Windows** サービスの名前と一致します。ユーザは、変化する可能性のあるプロセス ID (PID) またはサービス インデックスを指定するのではなく、プロセス名またはサービス名の正規表現を指定します。エージェントは、このユーザ指定の名前を使用して、モニタするプロセス (またはサービス) を検索します。デフォルトでは、プロセス モニタ テーブルは、プロセスまたはサービスが実行を停止した場合に、新しい PID またはサービスが見つかるまで、正規表現の適用を試行し続けます。

権限：読み書き

pmonMinValue

エージェントがプロセス属性のポーリングを開始してから観察した最低 (最小) 値を定義します。

権限：読み取り専用

pmonMaxValue

エージェントがプロセス属性のポーリングを開始してから観察した最高 (最大) 値を定義します。

権限：読み取り専用

pmonCurrentPID

現在モニタされているプロセスまたは **NT** サービス インデックスの **PID** を指定します。

権限：読み書き

pmonRowStatus

行ステータスを定義します。以下のいずれかになります。

- active
- notInService
- notReady
- createAndGo
- createAndWait
- destroy

通常、行は **active** または **notInService** のいずれかです。これらの値は、**SNMPv2** の **SMI RowStatus** テキスト表記法によって定義された値と意味が同一です。

権限：読み書き

デフォルト：**createAndWait(5)**

pmonNumEvents

このモニタ エントリのイベントの数を指定し、**pmonFlags** に依存します。イベントは必ずしもトラップを意味するとは限りません。

権限：読み書き

pmonObjClass

オブジェクト状態管理モデルで使用するために、モニタされる **MIB** オブジェクトのオブジェクトクラスを定義します。エージェントは、**sysedge.oid** ファイルを使用してデフォルト値を割り当てます。クラス名はモニタされたオブジェクト テーブルを指定するために使用できます。**SystemEDGE** が指定された名前を持つ既存のテーブルを決定しない場合、**SystemEDGE** はクラスを作成します。クラスはモニタリング オブジェクト識別子 (**SNMP OID**) を指定し、集計目的で参照できます。

重要：新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

pmonObjInstance

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト インスタンスを定義します。値が設定されていない場合、エージェントでデフォルトを判断できなければ、値は「不明」になります。表形式以外のオブジェクトのデフォルトは「NULL」です。リモート オブジェクトのシームレスなモニタリングの場合、インスタンス名はプレフィックス `//hostname/` で開始されます。

ローカル オブジェクトの場合は、エージェントがプレフィックス `//.` を各インスタンス名に追加します（「NULL」を含む）。この属性は、空文字列または「*」に設定することはできません。そのような要求は無視されます。インスタンス名が対応するインスタンスへの解析時に一致できない場合、**SystemEDGE** はこのモニタリング エントリの **OID** を指定されたインスタンス名で作成します。

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

pmonObjAttribute

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト インスタンスを定義します。エージェントは、`sysedge.oid` ファイルを使用して、MIB 属性名に対応するオブジェクトのデフォルト値を割り当てます。属性名がモニタリング エントリの解析時に不明な場合、**SystemEDGE** は指定された名前属性を作成します。

権限：読み書き

デフォルト：モニタされる **OID** の **ASCII MIB 属性名**

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

pmonSeverity

オブジェクト状態管理モデルに使用する重大度を定義します。この値は、しきい値違反が発生したときにエントリに割り当てる状態を定義します。有効な値は以下のとおりです。

- none(1)
- ok(2)
- warning(3)
- minor(4)
- major(5)
- critical(6)
- fatal(7)

none の値はレガシー モニタを作成し、エントリがオブジェクト状態管理モデルから除外されます。

デフォルト : **none**

pmonCurrState

エントリの取得された現在の状態を定義します。**pmonFlags** プロパティで定義された遅延値に基づいて、重大度および現在の違反の条件を使用して評価されます。有効な値は以下のとおりです。

- unknown(1)
- ok(2)
- warning(3)
- minor(4)
- major(5)
- critical(6)
- fatal(7)
- up(11)
- down(12)

up および down の値は、レガシー モニタ（重大度が none）に対してのみ表示されます。この列の状態変更は、状態が集計状態テーブル内の対応する管理対象オブジェクトのすべてのモニタで最悪である場合にのみ、状態変更トラップを送信します。

権限：読み取り専用

pmonProcName

正規表現によって一致したプロセス名を定義します。**Solaris 10** 以降で、ゾーン正規表現が使用される場合、名前の形式は **MatchedZoneName/MatchedProcName** となります。

サンプル プロセス モニタ テーブル エントリ

このセクションでは、httpd プロセスが稼働中であることを確認するために、その httpd プロセスをモニタするようにエージェントに指定する、サンプル プロセス モニタ テーブル エントリを示します。

Index	Desc	Interval	SampleType	Attribute
21	"Monitor httpd"	60	absoluteValue	procAlive
NumTraps	LastCall	Value	Oper	CurrVal
0	0d 22:40:00	4	=	3

LastTrap	RowStatus	Action	Flags	RegExpr
0	active	""	0x00000100	httpd
ObjClass	ObjInstance	ObjAttribute	Severity	CurrState
Process	httpd	procAlive	warning	ok

このサンプル エントリは以下の情報を提供します。

- この例におけるこのエントリは、プロセス モニタ テーブル内の 21 番目の行です。
- それは、60 秒ごとに Web サーバデーモン (httpd) をモニタするようにエージェントに指定します。
- モニタされている属性はプロセス (procAlive) のステータスです。
- 現在の値の 3 は、プロセスが (オペレーティング システムに関する限り) 正常に実行されていることを示します。プロセス ステータスが 4 に移動する場合、またはプロセスの実行が停止された場合、エージェントは現在の状態を warning (重大度の値によって定義) に設定し、このモニタが集計状態テーブル内の対応する管理対象オブジェクトの最悪の状態を表す場合には状態変更トラップを送信します。
- RowStatus 列は、このエントリがアクティブなことを示します。
- Action 列は NULL (") です。これは、プロセスの実行が停止されたか、再度開始されたときに、エージェントがアクションを呼び出さないことを示します。
- Flags 値 (0x00000100) は、子プロセスの代わりに、親 httpd デーモンをモニタするようにエージェントに指定します。フラグの詳細については、「[プロセス モニタ テーブルのフラグ \(P. 305\)](#)」を参照してください。
- クラス、インスタンス、および属性値は、集計状態テーブル内でプロセス モニタ エントリをこれらの値の管理対象オブジェクトの一部として確立します。さまざまな重大度をモニタするために他のプロセス モニタ エントリをこれらの同じ値で定義することや、このエントリを他の関連オブジェクトのほか、セルフ モニタ (同じプロセスの他の属性をモニタするなど) でも集計することができます。

プロセス属性

以下のリストでは、実行中（**procAlive**）の特定のプロセスまたはサービスに対して **SystemEDGE** エージェントがモニタできる属性および各属性の **SNMP** タイプについて説明します。**SNMP** タイプを使用して、サンプル タイプおよび演算子を選択できます。たとえば、**absoluteValue** サンプリングは、通常、**INTEGER** または **Gauge** の属性に最適で、**deltaValue** サンプリングは、通常、**Counter** 属性に最適です。

注：特定のプロセス属性をモニタするエージェントの機能は、関連するパラメータまたはメトリックを追跡する、基になるオペレーティング システムの機能に依存します。

procAlive (すべての OS)

プロセスまたはサービスが実行されているかどうかを定義します。

タイプ： **Boolean**

procMemory (すべての OS)

プロセスによって使用される実メモリのパーセンテージ (**0 ~ 100**) を定義します。

タイプ： **Gauge**

procSize (すべての OS)

テキスト、データ、およびスタックの各セグメントのサイズ (**KB** 単位) を定義します。

タイプ： **Gauge**

procRSS (すべての OS)

プロセスの実メモリ (常駐セット) のサイズ (**KB** 単位) を定義します。

タイプ： **Gauge**

procTime (すべての OS)

プロセスの累積された **CPU** 時間 (秒単位) を定義します。

タイプ： **Integer**

procInBlks (AIX、Solaris)

プロセスにより入力されたデータ ブロックの数を定義します。

タイプ : **Counter**

procOutBlks (AIX、Solaris)

プロセスにより出力されたデータ ブロックの数を定義します。

タイプ : **Counter**

procMsgsSent (AIX、HP-UX、Solaris)

プロセスによって受信されたメッセージの数を定義します。

タイプ : **Counter**

procMsgsRecv (AIX、HP-UX、Solaris)

プロセスによって送信されたメッセージの数を定義します。

タイプ : **Counter**

procNice (すべての OS)

プロセスの優先度 (**Nice** 値) を定義します。

タイプ : **Integer**

procNumThreads (Windows、AIX、Solaris、Linux)

このプロセスで実行されているスレッドの数を定義します。

タイプ : **Integer**

procNumSwaps (AIX、HP-UX、Solaris)

プロセスがスワップされた回数を定義します。

タイプ : **Counter**

procSysCalls (HP-UX、Solaris)

プロセスによって呼び出されたシステム コールの数を定義します。

タイプ : **Counter**

procMinorPgFlts (AIX、HP-UX、Linux、Solaris)

プロセスが原因で発生したマイナー ページ フォールトの数を定義します。

タイプ : **Counter**

procMajorPgFlts (すべての OS)

プロセスが原因で発生したメジャー ページ フォールトの数を定義します。

タイプ : **Counter**

procVolCtx (AIX、Solaris)

プロセスが原因で発生した任意コンテキスト切り替えの回数を定義します。

タイプ : **Counter**

procInvolCtx (AIX、Solaris)

プロセスが原因で発生した強制コンテキスト切り替えの回数を定義します。

タイプ : **Counter**

procTimePermil (すべての OS)

千分の一単位の CPU 使用率に係数 **1000** を掛けた値を定義します。

タイプ : **Integer**

行作成オブジェクト

プロセス モニタ テーブルの **monIndex** 列は、テーブル内の行を識別するための行インデックスとして機能します。行 1 ~ 10 は、**SystemEDGE** エージェントによる内部使用に予約済みです。行は 11 ~ **MAXINT** の範囲で設定できます。

プロセス モニタ テーブルを備えた以下の MIB オブジェクトを使用して、行作成を最適化することができます。

pmonUnusedIndex

オブジェクト上で **SNMP Get** を実行したときに、プロセス モニタ テーブルの未使用のインデックス番号を返します。行作成に使用できる番号を取得するには、このオブジェクトをクエリします。

pmonMatchDescr

pmonDescr 属性内の特定のエントリの説明に対応するインデックス番号を特定します。**pmonDescr** 属性値を指定してこの MIB オブジェクト上で **SNMP Set** を実行すると、エージェントがプロセス モニタ テーブル内のエントリを検索し、**monMatchIndex** MIB オブジェクト内で最後に一致したエントリのインデックス値を配置します。

pmonMatchIndex

pmonMatchDescr と共に使用したときに、特定のエントリの説明をそのインデックス番号と照合します。

ローカル ポリシーに基づいて、行のブロックをシステム管理者に予約するように選択することができます。その後、その行が別のユーザのエントリによってすでに取得されている場合があることを懸念せずに、行の予約済みブロック内のエントリを定義できます。その他のユーザがユーザ設定のエントリを定義する場合は、ローカル ポリシーに従って、予約された範囲外にある行インデックスを使用する必要があります。

行のブロックを予約することによって、一貫した条件のセット（行エントリ）がすべてのマシンにおいてモニタされるよう定義することが可能となり、各マシンの同じ行番号に同じ条件が定義されるようになります。たとえば、行 11

（monIndex = 11）を使用して **swapCapacity** 変数をモニタするためのエントリを定義することができます。また、この設定をすべてのシステムに配布できるため、行 11 はすべてのシステム上の **swapCapacity** 変数をモニタします。

CA Virtual Assurance でエージェントを設定すると、このプロセスを自動化できます。特定の行番号のエントリのブロックを設定し、1 回の操作ですべての管理対象システムにその設定を展開することができます。また、CA Virtual Assurance は、すでに使用されていて操作を拒否する行をユーザが定義しているかどうかを検出します。

CA Virtual Assurance 内での行の定義の詳細については、CA Virtual Assurance のドキュメントを参照してください。

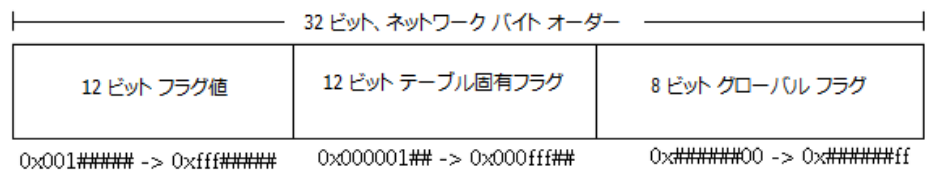
プロセス モニタ テーブルのフラグ

プロセス モニタ テーブル内の **pmonFlags** 列は 32 ビット符号なし整数フィールドで、対応する行の追加動作セマンティックを指定できます。

デフォルトでは、プロセス モニタ テーブル行は以下を実行します。

- それ自体を再初期化する試行
- SNMP トラップの送信（レガシー モニタのみ）
- 共通ログ ファイルへのイベントのログ記録
- アクションの呼び出し（アクションが設定されている場合）

これらのデフォルトを変更するように別のフラグ ビットを設定できます。エージェントは、すべてのフラグを 16 進（ベース 16）法で解釈します。以下の図は、プロセス モニタ テーブル フラグ（**pmonFlags**）フィールドの構成を示します。



フラグ値は 3 つのフィールドから構成されます。

- フィールド 1 には、**Systems Management Empire MIB** のモニタ テーブル用に定義された共通のテーブルフラグが含まれます。この部分はフラグの低位 8 ビットです。
- フィールド 2 には、各モニタ テーブル用に個別に定義された、テーブル固有のフラグが含まれます。このフィールドは、共通テーブルフラグの次の低位 12 ビットを定義します。
- フィールド 3 には、テーブル固有のフラグで使用する整数値用の予約済み高位 12 ビットが含まれます。このフィールドは、プロセス モニタ テーブルに固有のフラグを定義します。

以下のセクションでは、各フラグ ビットについて説明します。OR 論理演算によってフラグ値を組み合わせることができます。

以下のリストでは、プロセス モニタ テーブルのフラグについて説明します。

0x00000001

このエントリに対して、アクションが実行されないようにします。

0x00000002

このエントリに対して、**SNMP** トラップが送信されないようにします。

注: プロセス モニタ テーブルが制御するのは、レガシー モニタに対するトラップの送信のみです。ステートフルプロセス モニタについては、状態変更トラップが依然として集計状態テーブルから発生する場合があります。

0x00000004

このエントリを再初期化する試行を無効にします。デフォルトでは、エージェントは、ターゲットプロセスが再起動されている場合に、プロセス テーブルをスキャンして新しいプロセス ID を特定することにより、このエントリを定期的に再初期化しようとします。

0x00000008

sysedge.log ファイル内で、このエントリに対し、トラップのログが記録されないようにします。このビットを設定しても、トラップの送信には影響しません。イベントが頻繁に発生する場合、または特定のエントリをエージェントのハートビートとして使用している場合は、イベントのログ記録を無効にすると有用です。

0x00000010

エージェントがプロセスのモニタリングを再初期化しようとしてプロセスの一致に失敗するたびに、このエントリに対して継続的な **processStop** トラップを送信します。エージェントのデフォルト動作では、プロセスが停止したときに単一の **processStop** トラップが送信され、定期的にエントリの再初期化が試行されます。この機能を有効にすると、エージェントは、再初期化が失敗するたびに追加の **processStop** トラップを送信します。どのような場合でも、対応するエントリが **procAlive** プロセス属性をモニタしていない限り、エージェントは **processStart** と **processStop** のトラップを送信しません。

注: エージェントが **procAlive** 属性をモニタしているときに限り、このフラグは有効です。ユーザが Windows サービスをモニタしているときに、サービスが実行されていない場合、エントリは **notReady** 状態になりません。このフラグを設定すると、**SystemEDGE** は、サービスが実行されていないときに、準備完了状態のままだでも、**processStop** トラップを生成し、エントリの関連付けられた任意のアクションを実行します。

0x00000020

アクション スクリプトまたはプログラムへのデフォルト引数の受け渡しを無効にします。 **SystemEDGE** は、通常、トラップ タイプ、説明 フィールドなどを示すデフォルト アクション パラメータを渡します。アクション パラメータの詳細については、「[プロセス モニタ テーブルのアクション \(P. 312\)](#)」を参照してください。

0x00000040

このエントリに対して **notReady** トラップが送信されないようにします。**notReady** トラップのログは記録されず、**notReady** トラップに対してアクションも実行されません。

0x00000100

プロセス グループ内の親プロセスをモニタします。多くのアプリケーションおよびサービス（たとえば、**Web** サーバ **httpd** デーモン）は、最初のデーモンが子プロセスを生成して実際のサービス要求を処理するように設計されています。これらの子プロセスは、多くの場合、いくつかの要求を処理し、終了します。このような場合は、子プロセスの代わりに親デーモンをモニタするほうが望ましいといえます。この機能を有効にすると、エージェントは、プロセス正規表現の一致を検出した最初のプロセスの代わりに、親デーモンを検索およびモニタします。

エージェントは、プロセス正規表現（**pmonRegExpr**）の名前と一致するプロセスがないか、プロセス モニタ テーブルをスキャンすることによって、この検索を実行します。プロセスの一致する親プロセスも一致する場合、エージェントは親プロセスを返します。この検索アルゴリズムは親子関係のみに対応し、デーモンの分岐するデーモンを処理することはできません。この機能は **Windows** サービスのモニタリングには利用できません。

0x00000200

このエントリの **processStart** トラップの送信を無効にします（ログおよびアクションなし）。

0x00000400

プロセス モニタ 表現が **True** から **False** に移行するとき、このエントリに **processClear** トラップを送信します（ログおよびアクション）。この機能は、モニタされている属性が **procAlive** でないときのみ適用可能です。

0x00000800

モニタリングのプロセスをターゲットにするときに、プロセス名と引数が一致します。デフォルトでは、エージェントはプロセス名に対してのみ一致します。このオプションを有効にすると、エージェントはプロセス名およびプロセス引数の両方に **pmonRegExpr** を適用します。これは、同様のプロセス、あるいは同じアプリケーションまたはバイナリの複数の呼び出しを区別するために必要です。

0xXXX01000

processThreshold トラップを送信するか、*X* 回目の連続するイベント後初めて現在の状態を変更します。この機能を有効にすると、**processThreshold** トラップを送信するか、プロセス モニタの状態を変更する前に、*X* 回目の連続するイベントの発生まで待機するようにエージェントに指定されます。*X* 回目のイベントが発生した後、エージェントは後続の連続する **True** 表現評価ごとに **processThreshold** トラップを送信します。しきい値表現が **True** から **False** に再び移行すると、行はそれ自体をリセットし、エージェントはゼロからイベントを数え始めます。このフラグは、アクションの実行にも適用されます。フラグ値フィールドから *X* の値を指定できます。イベントのログ記録はこのフラグ ビットの影響を受けません。例については、「[プロセス モニタリングの例 \(P. 324\)](#)」を参照してください。

0xXXX02000

最大 *X* の連続する **processThreshold** トラップを送信し、それ以上送信しません。この機能を有効にすると、しきい値を超えたときに発生する可能性のある連続する **processThreshold** トラップおよびアクション実行の数の上限が設定されます。しきい値表現が **True** から **False** に移行すると、行はそれ自体をリセットし、エージェントはゼロからイベントを数え始めます。このフラグは、アクションの実行にも適用されます。フラグ値フィールドから *X* の値を指定できます。イベントのログ記録はこのフラグ ビットの影響を受けません。例については、「[プロセス モニタリングの例 \(P. 324\)](#)」を参照してください。

0xXXX04000

X プロセスがアライブかどうか、プロセス グループをモニタします (**procAlive** のモニタリングのみ)。

0x00008000

対応する正規表現に一致する **Windows** サービスをモニタします。このフラグを設定すると、**Windows** サービス テーブル内の一致する **Windows** サービスの **procAlive** 属性をモニタするように **SystemEDGE** に指定されます。

Solaris 10 上でゾーン内のプロセスをモニタします。 **pmonProcRegExpr** 属性の形式は **ZoneRegExpr/ProcRegExpr** であることが予期されます。

0x00010000

任意の **processStart** 処理を無効にします。このフラグが有効な場合、**processStart** イベントが発生すると、**SystemEDGE** は、アクションの呼び出し、イベントのログ記録、トラップの送信を行いません（レガシーサポート用の **000 002 00** も参照）。

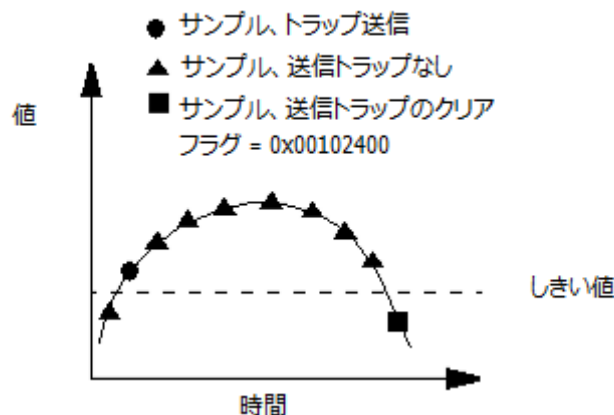
0x00020000

オブジェクトおよび重大度が同じモニタの状態を集計するため、モニタには **AND** 関係があります。このフラグセットがある同じ重大度の接続されたすべてのモニタには、その都度変化する集計状態に対する違反条件が必要です。

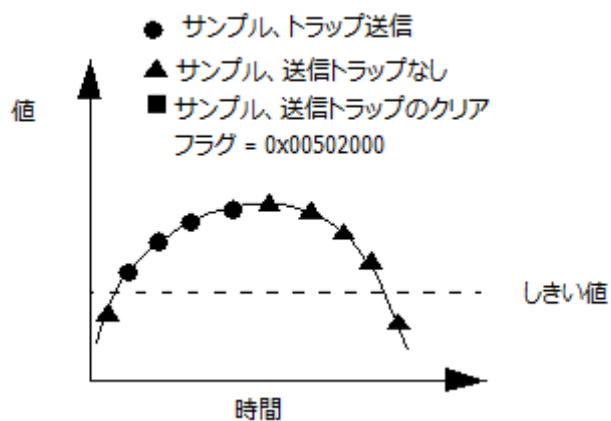
0x###00000

いくつかのフラグ ビットでは、トラップを送信し、アクションを実行するために、値 **X** が使用されます。値 **X** はフラグ フィールドの高順位の 12 ビットとして指定されます。このフィールドを利用するフラグ ビットは相互に排他的です。

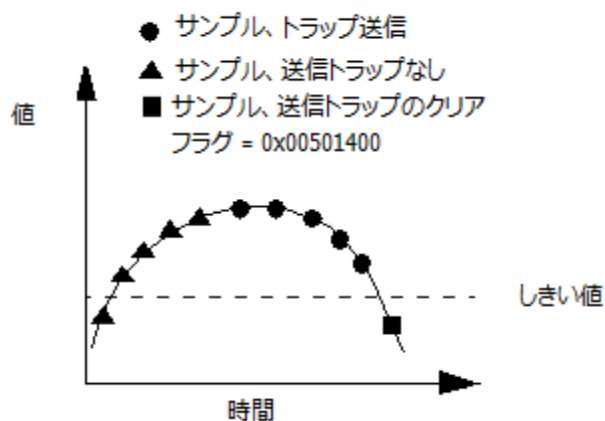
以下の図は、モニタされるオブジェクトがしきい値を超えていることを示すために 1 つのトラップを送信し、値がしきい値を下回ったときには **processClear** トラップを送信するエージェントを示します。



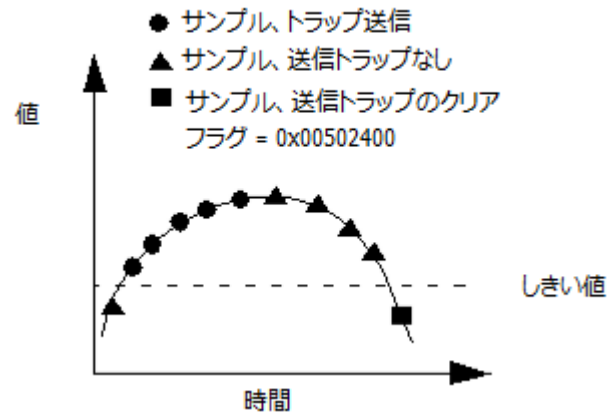
以下の図は、モニタされるオブジェクトの値がしきい値を超えていることを示すために 4 つのトラップを送信するエージェントを示します。値がしきい値を下回ると、processClear トラップは送信されません。



以下の図は、トラップの送信を開始する前にイベントが複数回発生するまで待機するエージェントを示します。モニタされるオブジェクトの値がしきい値を下回るまでトラップが送信され、しきい値を下回ると processClear トラップが送信されます。



以下の図は、指定された回数だけトラップを送信するエージェントを示します。モニタされるオブジェクトの値がしきい値を下回ると、エージェントは processClear トラップを送信します。



プロセス モニタ テーブルのアクション パラメータ

アクション コマンドが呼び出されると、SystemEDGE エージェントはそれらのアクション コマンドにいくつかのデフォルト パラメータを渡します。これらのパラメータは、アクション文字列で指定したパラメータに加えて使用され、指定したパラメータの後コマンドラインに渡されます。デフォルト パラメータは、プロセス モニタ テーブルに対して送信される SNMP トラップ内に用意されているパラメータと同じです。

以下のリストでは、プロセス モニタ テーブルのアクションのデフォルト パラメータについて説明します。

注: テーブル エントリで 0x00000020 フラグを設定することによって、エージェントがこれらのアクション パラメータをコマンドに含めないようにすることができます。

trapType

送信されているトラップのタイプを定義します。それは以下のいずれかになります。

- processStop
- processStart
- processThreshold
- processClear

pmonIndex

テーブル エントリに割り当てられたインデックスを定義します。

pmonDescr

テーブル エントリの説明を定義します。

pmonAttribute

エントリによってモニタされているプロセス属性を定義します。

pmonCurrVal

テーブル エントリに対して取得された現在の値を定義します。

pmonOperator

テーブル エントリによって使用されている演算子を定義します。

pmonValue

テーブル エントリによって適用された比較値またはしきい値を定義します。

pmonFlags

テーブル エントリと関連付けられたフラグを定義します。

pmonRegExpr

モニタするプロセスのプロセス ID を検索するためにエージェントが使用する正規表現を定義します。

pmonCurrentPID

モニタされている現在のプロセスのプロセス ID (PID) を定義します。ユーザが **Windows** サービスを (**watch ntservice** ディレクティブまたはフラグ **0x8000** の設定によって) モニタしている場合、この列は、エントリをモニタするこのプロセスまたはサービスが取得した **NT Service MIB** テーブル内のインデックスを表します。

pmonObjClass

オブジェクト状態管理モデルで使用するために、モニタされる **MIB** オブジェクトのオブジェクトクラスを定義します。エージェントは、**sysedge.oid** ファイルを使用してデフォルト値を割り当てます。クラス名はモニタされたオブジェクト テーブルを指定するために使用できます。**SystemEDGE** が指定された名前を持つ既存のテーブルを決定しない場合、**SystemEDGE** はクラスを作成します。クラスはモニタリングオブジェクト識別子 (**SNMP OID**) を指定し、集計目的で参照できます。

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

pmonObjInstance

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト インスタンスを定義します。値が設定されていない場合、エージェントでデフォルトを判断できなければ、値は「不明」になります。表形式以外のオブジェクトのデフォルトは「NULL」です。リモートオブジェクトのシームレスなモニタリングの場合、インスタンス名はプレフィックス `//hostname/` で開始されます。

ローカルオブジェクトの場合は、エージェントがプレフィックス `//.` を各インスタンス名に追加します（「NULL」を含む）。この属性は、空文字列または「*」に設定することはできません。そのような要求は無視されます。インスタンス名が対応するインスタンスへの解析時に一致できない場合、**SystemEDGE** はこのモニタリング エントリの **OID** を指定されたインスタンス名で作成します。

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

pmonObjAttribute

オブジェクト状態管理モデルで使用するために、モニタされる MIB オブジェクトのオブジェクト インスタンスを定義します。エージェントは、**sysedge.oid** ファイルを使用して、MIB 属性名に対応するオブジェクトのデフォルト値を割り当てます。属性名がモニタリング エントリの解析時に不明な場合、**SystemEDGE** は指定された名前属性を作成します。

権限：読み書き

デフォルト：モニタされる **OID** の **ASCII MIB 属性名**

重要: 新たに作成された参照は、モニタリング エントリが既存の **SNMP OID** に解決される場合のみ有効です。

pmonCurrState

テーブル エントリの現在の状態を定義します。

SystemEDGE エージェントによって送信されるトラップの詳細については、「コンセプト」の章および付録「プライベート企業トラップ」を参照してください。

プロセス モニタ テーブルの表示

CA Virtual Assurance を使用して、管理対象エージェントのプロセス モニタ テーブルの内容を表示することができます。

CA Virtual Assurance でプロセス モニタ テーブルを表示する方法

1. CA Virtual Assurance ユーザ インターフェイスにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペイン内の [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバの詳細のページが表示されます。
4. [設定] をクリックし、[プロセス モニタ] をクリックします。
プロセス モニタ テーブルが、すべての既存のプロセス モニタと共に表示されます。

プロセス モニタリングの設定

プロセス モニタ テーブルでエントリを追加、削除、または変更することによって、SystemEDGE エージェントがモニタするプロセスとプロセス属性を制御することができます。

プロセス モニタ テーブルは以下の方法で設定できます。

CA Virtual Assurance からのファイルベース設定の動的使用

CA Virtual Assurance では、管理しているシステムにインストールされているエージェントを **CA Virtual Assurance** ユーザ インターフェースから管理できます。エージェントの再起動が必要ないユーザ フレンドリーで安全なメカニズムを備えたモニタを動的に定義して送信する機能が備わっているため、これは設定の推奨された方法です。個々のエージェントのモニタを設定するか、または複数の管理対象システムで適用するモニタ ポリシーを定義できます。個々の設定を変更するか、設定ポリシーを適用すると、**CA Virtual Assurance** によりエージェントシステムに新しい設定ファイルが送信されます。**CA Virtual Assurance** は、データ センター全体におけるエージェント設定の中心点の役割を果たします。また、他の方法によって加えられた変更を上書きしたり、拒否することができます。

CA Virtual Assurance を使用したエージェントの設定の詳細については、**CA Virtual Assurance** のドキュメントを参照してください。

eHealth AdvantEDGE View、CA Spectrum、または CA NSM などの管理システムからの SNMP Set コマンドの動的使用

さまざまな管理システムで、**SNMP Set** コマンドによる **SystemEDGE** へのモニタの追加がサポートされます。**eHealth** では、**AdvantEDGE View** インターフェースを介してこの機能を利用できます。また、**CA NSM** ではエージェント ビューを介して設定が可能で、**CA Spectrum** では **OneClick** を介して設定が可能です。**CA Virtual Assurance** を介した動的設定により利用可能な機能はどれも、これらの方法では利用できません。

SNMP Set の設定の詳細については、該当する管理システム用のドキュメントを参照してください。

ローカル sysedge.cf ファイルでのセルフ モニタリング テーブルのエントリの手動指定

マニュアル設定では、特定のディレクティブを使用してローカル **sysedge.cf** ファイルにエントリを追加する必要があります。**sysedge.cf** ファイルからエージェントを直接設定した場合、変更を適用するには完全な再起動が必要です。直接設定するにはエージェント データ ディレクトリにある **sysedge.cf** ファイルを使用します。

このセクションでは、ローカル **sysedge.cf** ファイルでセルフ モニタリングを直接設定する方法について説明しますが、あらゆる種類の設定に必要なプロパティについて説明します。

edgework ユーティリティを介して、プロセス モニタ テーブル エントリを動的に追加、削除、または修正することもできます。詳細については、「edgework ユーティリティ -- プロセスのモニタ」を参照してください。

watch process ディレクティブ -- プロセス モニタ テーブルへのエントリの追加

watch process ディレクティブを使用すると、sysedge.cf ファイルでプロセス モニタ テーブルにプロセス モニタ エントリを直接追加できます。引数はプロセス モニタ テーブルの列を表します。

以下に記述する構文を使用してエージェント データ ディレクトリ内の sysedge.cf に行を追加し、ファイルを保存し、変更を有効にするためにエージェントを再起動します。

procAlive など、利用可能な任意のプロセス属性をモニタするには、この構文を使用します。

プロセスが動作しているかどうかをモニタする属性として procAlive を入力すると、特定の値が無視されます。表現「hrSWRunStatus=4」に基づいて procAlive 属性モニタをモニタします。プロセスが実行を停止したか無効になった場合、エージェントは processStop (レガシー モニタ) または状態変更トラップ (ステートフルモニタ) を送信し、任意の設定アクションを呼び出します。

0x00008000 フラグを使用して Windows サービスをモニタするように指定すると、同じモニタは無視され、エージェントは表現「ntServiceState ?1」に基づいてサービスの procAlive 属性をモニタします。サービスが実行を停止したか無効になると、エージェントは processStop (レガシー モニタ) または状態変更トラップ (ステートフルモニタ) を送信し、任意の設定アクションを呼び出します。

watch process ディレクティブを使用して、プロセス モニタ テーブルにエントリを以下のように追加します。

```
watch process attribute 'regexpr' index flags interval stype oper thresh ['descr'
['action' ['objClass' 'objInst' 'objAttr' severity]]]
```

attribute

指定されたしきい値を SystemEDGE エージェントがモニタするプロセス属性を指定します。procAlive など、「[プロセス属性 \(P. 301\)](#)」のどんなプロセス属性でも指定できます。

'regexpr'

モニタするアプリケーションまたはプロセスのプロセス ID を取得しようとするときに使用する正規表現を指定します。モニタするプロセス名の正規表現を指定してください。

index

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 **1 ～ 10** はエージェントによる内部使用に予約済みです。したがって、インデックス値は **10** より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば **0x00000001**）を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグの詳細については、「[プロセス モニタ テーブルのフラグ \(P. 305\)](#)」を参照してください。

interval

モニタを実行する必要がある頻度を秒単位で指定します。たとえば、値 **30** は、このエントリを **30** 秒ごとにモニタするようにエージェントに指示します。最小値は **30** で、**30** 秒の倍数で指定する必要があります。

stype

エージェントが属性の絶対値（**absoluteValue**）をサンプリングすべきか、または連続するサンプルの差異（**deltaValue**）をとるべきかを指定します。変化のレートを知るためにカウンタ変数をモニタするには、**deltaValue** を使用します。正確な値を知るためにゲージをモニタするには、**absoluteValue** を使用します。

注: ユーザが **procAlive** 属性をモニタしている場合、エージェントはこの値を無視します。

oper

現在のエントリ値を提供されたしきい値と比較評価するためのブール演算子を指定します。

演算子として有効な値は、以下のとおりです。

- nop（演算なし）
- >（より大きい）
- <（より小さい）
- >=（以上）
- <=（以下）
- =（等しい）
- !=（等しくない）

デフォルト：**nop**

注：ユーザが **procAlive** 属性をモニタしている場合、エージェントはこの値を無視します。

thresh

指定された演算子を使用して、モニタ対象属性の現在値と比較される整数しきい値を指定します。

注：ユーザが **procAlive** 属性をモニタしている場合、エージェントはこの値を無視します。

'descr'

テーブルエントリの任意の説明（0 ～ 512 文字の長さ）を指定します。

'action'

しきい値条件が満たされ、エージェントがしきい値違反トラップ（レガシーモニタ）または状態変更トラップ（ステートフルモニタ）を送信したときに実行されるコマンド（長さが 0 ～ 2048 文字）を、フルパスおよび任意のパラメータを含めて指定します。ステートフルプロセスモニタについては、しきい値違反がオブジェクトの状態変更をもたらさない場合、コマンドは実行されません。文字列が空の場合、エージェントはこのエントリに対してアクションを実行しません。

注：エージェントがアクションを実行するときのデフォルト設定は変更できません。詳細については、「エージェントの設定」の章を参照してください。

'objClass'

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクトクラスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を（**objInst** および **objAttr** と共に）定義します。

デフォルト： **MIB** グループまたはテーブル名（「**sysedge.oid**」を参照）

'objInst'

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクトインスタンスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を（**objClass** および **objAttr** と共に）定義します。エージェントは、デフォルト値として **specified regExpr** プロパティの値を使用します。

'objAttr'

状態管理モデルで使用するためにオブジェクト属性を指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を（**objClass** および **objInst** と共に）定義します。

デフォルト： プロセス属性名

severity

オブジェクト状態管理モデルに使用する重大度を指定します。この値は、しきい値違反が発生したときにエントリに割り当てる状態を定義します。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

none の値はレガシー モニタを作成し、エントリがオブジェクト状態管理モデルから除外されます。

デフォルト： なし

watch ntservice ディレクティブ -- プロセス モニタ テーブルへのサービス モニタリング エントリの追加

watch ntservice ディレクティブを使用すると、実行ステータスを追跡するために procAlive 属性に基づいて Windows サービスをモニタするためのエントリを、プロセス モニタ テーブルに追加できます。

以下に記述する構文を使用してエージェント データ ディレクトリ内の sysedge.cf に行を追加し、ファイルを保存し、変更を有効にするためにエージェントを再起動します。

watch ntservice ディレクティブを使用して、プロセス モニタ テーブルに Windows サービス モニタリング エントリを以下のように追加します。

```
watch ntservice 'regexpr' index flags interval ['descr' ['action' ['objClass' 'objInst' 'objAttr' severity]]]
```

regexpr

モニタするサービスの ID を取得しようとするときに使用する正規表現を指定します。モニタするサービス名の正規表現を指定してください。

index

このエントリに使用するモニタ テーブルの行 (インデックス) を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 1 ~ 10 はエージェントによる内部使用に予約済みです。したがって、インデックス値は 10 より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値 (たとえば 0x00000001) を使用して、このエントリの動作に関する追加の指示を指定します。

interval

モニタを実行する必要がある頻度を秒単位で指定します。たとえば、値 30 は、このエントリを 30 秒ごとにモニタするようにエージェントに指示します。最小値は 30 で、30 秒の倍数で指定する必要があります。

'descr'

テーブル エントリの任意の説明 (0 ~ 512 文字の長さ) を指定します。

'action'

しきい値条件が満たされ、エージェントがしきい値違反トラップ（レガシー モニタ）または状態変更トラップ（ステートフル モニタ）を送信したときに実行されるコマンド（長さが 0 ～ 2048 文字）を、フルパスおよび任意のパラメータを含めて指定します。ステートフルプロセス モニタについては、しきい値違反がオブジェクトの状態変更をもたらさない場合、コマンドは実行されません。文字列が空の場合、エージェントはこのエントリに対してアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定は変更できます。詳細については、「エージェントの設定」の章を参照してください。

'objClass'

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクトクラスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を（**objInst** および **objAttr** と共に）定義します。

デフォルト: **processEntry**

'objInst'

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクトインスタンスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を（**objClass** および **objAttr** と共に）定義します。エージェントは、デフォルト値として **specified regExpr** プロパティの値を使用します。

'objAttr'

状態管理モデルで使用するためにオブジェクト属性を指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を（**objClass** および **objInst** と共に）定義します。

デフォルト: **procAlive**

severity

オブジェクト状態管理モデルに使用する重大度を指定します。この値は、しきい値違反が発生したときにエントリに割り当てる状態を定義します。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

`none` の値はレガシー モニタを作成し、エントリがオブジェクト状態管理モデルから除外されます。

デフォルト： なし

プロセス モニタリングの例

このセクションには、プロセス モニタリングのサンプル設定ファイルディレクティブが含まれています。例ではそれぞれ、プロセス モニタリングのインスタンスを定義する方法を示し、モニタされている属性またはしきい値について説明します。

例: sendmail が実行されていることを確認するための sendmail のモニタ

以下の例では、UNIX 電子メール サーバの基盤となるシステム上で sendmail デーモンをモニタするようにエージェントを設定します。

```
watch process procAlive 'sendmail' 11 0x00000100 60 absolute=4 'Monitor sendmail' ''  
'processEntry' 'sendmail' 'procAlive' fatal
```

11

このエントリがプロセス モニタ テーブルの行 11 (**pmonIndex=11**) を占めることを示します。

0x00000100

複数の **sendmail** デーモンが存在し実行されている場合に、エージェントが親 **sendmail** プロセスをモニタすることを示します。

60

エージェントが **60** 秒ごとに **sendmail** プロセスを確認することを示します。

processEntry

このエントリでオブジェクト状態モデルの **processEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

sendmail

このエントリでオブジェクト状態モデルの **sendmail** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/」は無効な値です。

procAlive

このエントリでオブジェクト状態モデルの **procAlive** オブジェクト属性が使用されることを示します。属性はデフォルト値に置き換えられます。ここで、「*」は無効な値です。

fatal

このエントリでオブジェクト状態モデルの **fatal** 重大度が使用されることを示します。

アクションは指定されないため、エージェントはトラップの送信時にコマンドを呼び出しません。

例: 簡易 TCP/IP サービスのプロセスが実行されていることを確認するための簡易 TCP/IP サービスのプロセスのモニタ

以下の例では、簡易 TCP/IP サービスのサービスを構成する TCPSVCS プロセスをモニタするようにエージェントを設定します。

```
watch process procAlive 'TCPSVCS' 12 0x00000000 30 absolute=4 'Monitor NT TCP
services' '' 'processEntry' 'TCPSVCS' 'procAlive' warning
```

12

このエントリがプロセス モニタ テーブルの行 **12** (**pmonIndex=12**) を占めることを示します。

0x00000000

エージェントがデフォルトプロセスのモニタリング動作を実行することを示します。

30

エージェントが **30** 秒ごとに **TCPSVCS** プロセスを確認することを示します。

processEntry

このエントリでオブジェクト状態モデルの **processEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

TCPSVCS

このエントリでオブジェクト状態モデルの **TCPSVCS** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/」は無効な値です。

procAlive

このエントリでオブジェクト状態モデルの **procAlive** オブジェクト属性が使用されることを示します。属性はデフォルト値に置き換えられます。ここで、「*」は無効な値です。

warning

このエントリでオブジェクト状態モデルの **warning** 重大度が使用されることを示します。

アクションは指定されないため、エージェントはトラップの送信時にコマンドを呼び出しません。

注: この例では、Windows 簡易 TCP/IP サービスのサービスを提供する基盤となるプロセスをモニタする方法を示します。以下の例では、その基盤となるプロセスの代わりに Windows サービス自体をモニタする方法を示します。

例: 簡易 TCP/IP サービスのサービスのモニタ

以下の例は両方とも、基盤となるプロセスの代わりに TCPSVCS サービス自体をモニタするようにエージェントを設定します。

```
watch process procAlive 'Simple TCP/IP Services' 13 0x08000 30 absolute=4 'Monitor  
NT TCP/IP Services' '' 'processEntry' 'Simple TCP/IP Services' 'procAlive' minor  
13
```

このエントリがプロセス モニタ テーブルの行 13 (**pmonIndex=13**) を占めることを示します。

0x08000

エージェントが基盤となるプロセスの代わりに **Windows** サービスをモニタするとを示します。

30

エージェントが **30** 秒ごとに簡易 TCP/IP サービスのサービスを確認することを示します。

processEntry

このエントリでオブジェクト状態モデルの **processEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

Simple TCP/IP Services

このエントリでオブジェクト状態モデルの簡易 TCP/IP サービス オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*instance」、および「//*/*」は、無効な値です。

procAlive

このエントリでオブジェクト状態モデルの **procAlive** オブジェクト属性が使用されることを示します。属性はデフォルト値に置き換えられます。ここで、「*」は無効な値です。

minor

このエントリでオブジェクト状態モデルの **minor** 重大度が使用されることを示します。

```
watch ntservice 'Simple TCP/IP Services' 14 0x0 30 'Monitor NT TCP/IP Services' ''  
'processEntry' 'Simple TCP/IP Services' 'procAlive' major
```

14

このエントリがプロセス モニタ テーブルの行 14 (**pmonIndex=14**) を占めることを示します。

30

エージェントが **30** 秒ごとに簡易 **TCP/IP** サービスのサービスを確認することを示します。

processEntry

このエントリでオブジェクト状態モデルの **processEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

Simple TCP/IP Services

このエントリでオブジェクト状態モデルの簡易 **TCP/IP** サービス オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/*」は、無効な値です。

procAlive

このエントリでオブジェクト状態モデルの **procAlive** オブジェクト属性が使用されることを示します。属性はデフォルト値に置き換えられます。ここで、「*」は無効な値です。

major

このエントリでオブジェクト状態モデルの **major** 重大度が使用されることを示します。

アクションは指定されないため、エージェントはトラップの送信時にコマンドを呼び出しません。

例: sendmail が実行されていることを確認するための ypbind のモニタ

以下の例では、基盤となるシステム上で UNIX ypbind デーモンをモニタするようにエージェントを設定します。

```
watch process procAlive 'ypbind' 15 0x00000000 60 absolute=4 'Monitor ypbind'  
'/example/pager.sh' 'processEntry' 'ypbind' 'procAlive' critical
```

15

このエントリがプロセス モニタ テーブルの行 15 (**pmonIndex=15**) を占めることを示します。

60

エージェントが 60 秒ごとに **ypbind** プロセスを確認することを示します。

processEntry

このエントリでオブジェクト状態モデルの **processEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

ypbind

このエントリでオブジェクト状態モデルの **ypbind** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/」は、無効な値です。

procAlive

このエントリでオブジェクト状態モデルの **procAlive** オブジェクト属性が使用されることを示します。属性はデフォルト値に置き換えられます。ここで、「*」は無効な値です。

critical

このエントリでオブジェクト状態モデルの **critical** 重大度が使用されることを示します。

エージェントは、トラップを送信するたびに、指定されたアクション スクリプト `/example/pager.sh` を呼び出します。この場合、**processStop** または **processStart** トラップが送信されるたびに、エージェントはスクリプトを起動します。スクリプトは、その引数を検査して送信されているトラップを特定し、ターゲット ポケットベルに適切なメッセージを送信します。

例: プロセスのサイズのモニタ

以下の例では、特定のプロセス全体のサイズをモニタするようにエージェントを設定します。

```
watch process procSize 'netscape' 20 0x00a02400 60 absolute '>' 35000 'Monitor  
netscape size' '' 'processEntry' 'netscape' 'procSize' none
```

procSize

エージェントがモニタしている属性を示します。対応するプロセスのテキスト、データ、およびスタック セグメントのサイズを返します。指定されたしきい値に対してこの属性をモニタすることで、これがメモリ リークを発生させているかどうかを判断できます。

netscape

エージェントがモニタするプロセスの名前を示します。

20

このエントリがプロセス モニタ テーブルの行 **20** (**pmonIndex=20**) を占めることを示します。

0x00a02400

デフォルトのプロセス モニタ テーブルの動作を以下のように変更するようにエージェントに指定します。

0x00000400

processClear トラップを送信するようにエージェントに指定します。

0x00002000

最大 **10** の連続するトラップを送信し、それ以上送信しないようにエージェントに指定します。

0x00a00000

この例のディレクティブで使用するために、フラグ値 **10** が含まれます。

processEntry

このエントリでオブジェクト状態モデルの **processEntry** オブジェクトクラスが使用されることを示します。オブジェクトクラスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

netscape

このエントリでオブジェクト状態モデルの **netscape** オブジェクトインスタンスが使用されることを示します。オブジェクトインスタンスはデフォルト値に置き換えられます。ここで、「*」は無効な値です。

注: 「//」および「/」は、「//hostname/instance」のように、システムにインスタンスを関連付けるための区切り文字として使用できます。ローカルシステムインスタンスの場合は、ローカルシステムにインスタンスを関連付けるために「//./」が常に先頭に付けられます。「//hostname/*」、「//*/instance」、および「//*/」は、無効な値です。

procAlive

このエントリでオブジェクト状態モデルの **procAlive** オブジェクト属性が使用されることを示します。属性はデフォルト値に置き換えられます。ここで、「*」は無効な値です。

none

このエントリでオブジェクト状態モデルの **none** 重大度が使用されることを示します。

absolute

連続するサンプル間の差異（デルタ）を測定するのではなく、サンプリングされた値をしきい値と比較するようにエージェントに指定します。

>

サンプリングされた **procSize** 属性を値 **35000**（35,000 KB または **35 MB**）と比較し、しきい値を超えたときは **processThreshold** トラップを送信するようにエージェントに指定します。

関連項目

[Perl 互換正規表現（PCRE）のサポート](#) (P. 178)

[テキスト パターン除外の設定](#) (P. 181)

[正規表現の例](#) (P. 181)

プロセス モニタリング用の edgework コマンド

edgework プロセス モニタリング コマンドおよび関連する引数は以下のとおりです。

```
add procAlive processname index flags interval 'descr' 'action' [objClass objInst objAttr severity]
```

```
add attribute processname index flags interval sampleType oper value 'description' 'action' [objClass objInst objAttr severity]
```

```
setStatus index status
```

```
delete index
```

```
list
```

```
dump
```

processname

モニタするプロセスの **PID** の検索に使用される正規表現を指定します。スペースまたは他の特殊文字が含まれる場合は、この値を引用符で囲みます。

index

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 **1 ~ 10** はエージェントによる内部使用に予約済みです。したがって、インデックス値は **10** より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば **0x00000001**）を使用して、このエントリの動作に関する追加の指示を指定します。

interval

モニタを実行する必要がある頻度を秒単位で指定します。たとえば、値 **30** は、このエントリを **30** 秒ごとにモニタするようにエージェントに指示します。最小値は **30** で、**30** 秒の倍数で指定する必要があります。

sampleType

絶対値またはデルタのいずれかのサンプル タイプを指定します。この値は、エージェントがオブジェクトの絶対値をサンプリングするか、連続するサンプル間の差異を取得するかを示します。

oper

現在のエントリ値を指定されたしきい値と突き合わせて評価するためのブール演算子を指定します。

演算子の有効な値は以下のとおりです。

- nop (演算なし)
- > (より大きい)
- < (より小さい)
- >= (以上)
- <= (以下)
- = (等しい)
- != (等しくない)
- デフォルト: nop

value

モニタされる属性の現在値が指定された演算子を使用して比較される整数しきい値を指定します。

'descr'

テーブルエントリの任意の説明 (長さが 0 ～ 512 文字) を指定します。

'action'

しきい値条件が満たされ、エージェントがしきい値違反トラップ (レガシー モニタ) または状態変更トラップ (ステートフル モニタ) を送信したときに実行されるコマンド (長さが 0 ～ 2048 文字) を、フルパスおよび任意のパラメータを含めて指定します。ステートフルセルフモニタについては、しきい値違反がオブジェクトの状態変更をもたらさない場合、コマンドは実行されません。文字列が空の場合、エージェントはこのエントリに対してアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定は変更できます。詳細については、「エージェントの設定」の章を参照してください。

objClass

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクトクラスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を (**objInst** および **objAttr** と共に) 定義します。

デフォルト: **processEntry**

objInst

状態管理モデルで使用するために、モニタされるオブジェクトのオブジェクトインスタンスを指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を (**objClass** および **objAttr** と共に) 定義します。エージェントは、デフォルト値として **specified regExpr** プロパティの値を使用します。

objAttr

状態管理モデルで使用するためにオブジェクト属性を指定します。値が同じすべてのエントリを集計する管理対象オブジェクトを作成するために、このプロパティの値を (**objClass** および **objInst** と共に) 定義します。

デフォルト: プロセス属性名

severity

オブジェクト状態管理モデルに使用する重大度を指定します。この値は、しきい値違反が発生したときにエントリに割り当てる状態を定義します。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

none の値はレガシー モニタを作成し、エントリがオブジェクト状態管理モデルから除外されます。

デフォルト: なし

status

エントリのステータスを定義します。エントリのステータスは以下のいずれかになります。

- active (行をアクティブ化する)
- notInService (行を非アクティブ化するが、保持する)
- destroy (セルフ モニタ テーブル行を削除する)

attribute

指定されたしきい値をエージェントがモニタするプロセス属性を指定します。「[プロセス属性 \(P. 301\)](#)」のリストのどんなプロセス属性でも選択できます。

edgewatch の例

このセクションでは、SNMP バージョン 1、2c、および 3 で edgewatch コマンドを使用する例を示します。

例: ypbind プロセスのモニタ

以下の例では、SNMPv1 のターゲット システム上で実行される ypbind プロセスをモニタするプロセス モニタ テーブル エントリを、インデックス 16 で作成します。

```
edgewatch -h 143.45.0.12 -c private -v 1 -o process add procAlive "ypbind" 16 0x00
60 "Monitor ypbind" "/example/pager.sh" "" "" "" minor
```

以下の例では、SNMPv2c に対して同じ設定を作成します。

```
edgewatch -h 143.45.0.12 -c private -v 2c -o process add procAlive "ypbind" 16 0x00
60 "Monitor ypbind" "/example/pager.sh" "" "" "" fatal
```

以下の例では、SNMPv3 に対して同じ設定を作成します。

```
edgewatch -h 143.45.0.12 -v 3 -u userName -s 3 -a authPassword -A MD5 -x encryptPassword
-X DES -o process add procAlive "ypbind" 16 0x00 60 "Monitor ypbind"
"/example/pager.sh" "" "" "" major
```

procAlive

モニタされているプロセス属性を示します。プロセスが実行されていることを確認するためにプロセスをモニタするようにエージェントに指定します。ypbind はエージェントがモニタしているプロセスです。ypbind は、クライアントディレクトリのルックアップを実行し、NIS (Network Information Services) を実行するコンピュータに必要です。

0x00

このテーブル エントリのデフォルトの動作を実行するようにエージェントに指定します。

プロセスが停止した場合、エージェントは processStop トラップを送信し、アクション スクリプト /example/pager.sh を実行します。

例: firefox プロセスのモニタ

以下の例では、SNMPv1 のターゲット コンピュータ上で実行される **firefox** プロセスをモニタするプロセス モニタ テーブル エントリを、インデックス 20 で作成します。

```
edgwatch -h 143.45.0.12 -c private -v 1 -o process add procSize "firefox" 20
0x00a02400 60 absolute ">" 35000 "Monitor firefox size" "" "" "" "" critical
```

以下の例では、SNMPv2c に対して同じ設定を作成します。

```
edgwatch -h 143.45.0.12 -c private -v 2c -o process add procSize "firefox" 20
0x00a02400 60 absolute ">" 35000 "Monitor firefox size" "" "" "" "" none
```

以下の例では、SNMPv3 に対して同じ設定を作成します。

```
edgwatch -h 143.45.0.12 -v 3 -u userName -s 3 -a authPassword -A MD5 -x encryptPassword
-X DES -o process add procSize "firefox" 20 0x00a02400 60 absolute ">" 35000 "Monitor
firefox size" "" "" "" "" ok
```

procSize

モニタされているプロセス属性を示します。プログラムのテキスト、データ、およびスタック セグメント全体のサイズをモニタするようにエージェントに指定します。

netscape

エージェントがモニタしているアプリケーションを示します。

0x00a02400

デフォルトのプロセス モニタ テーブルの動作を以下のように変更するようにエージェントに指定します。

0x00000400

processClear トラップを送信するようにエージェントに指定します。

0x00002000

最大 **10** の連続するトラップを送信し、それ以上送信しないようにエージェントに指定します。

0x00a00000

このディレクティブで使用するために、フラグ値 **10** が含まれます。

>

Netscape のプロセス サイズがしきい値 (**35 MB**) を越えると、イベントが発生することを示します。

35,000 KB または 35 MB

しきい値を示します。

例: Windows TCPSVCS プロセスのモニタ

以下の例では、SNMPv1 のターゲット システム上で実行される Windows TCPSVCS プロセス (またはサービス) をモニタするプロセス モニタ テーブル エントリを、インデックス 15 で作成します。

```
edgewatch -h 143.45.0.12 -c private -v 1 -o process add procAlive "TCPSVCS" 15 0x00
30 "Monitor NT TCP services" "" "" "" "" fatal
```

以下の例では、SNMPv2c に対して同じ設定を作成します。

```
edgewatch -h 143.45.0.12 -c private -v 2c -o process add procAlive "TCPSVCS" 15 0x00
30 "Monitor NT TCP services" "" "" "" "" ok
```

以下の例では、SNMPv3 に対して同じ設定を作成します。

```
edgewatch -h 143.45.0.12 -v 3 -u userName -s 3 -a authPassword -A MD5 -x encryptPassword
-X DES -o process add procAlive "TCPSVCS" 15 0x00 30 "Monitor NT TCP services" "" ""
"" "" warning
```

procAlive

モニタされているプロセス属性を示します。このプロセスが実行されていることを確認するためにプロセス モニタ テーブルを定期的に (**30** 秒ごとに) スキャンするようにエージェントに指定します。

TCPSVCS

Windows システム上の **TCP** 関連のサービスを実行する **Windows** サービスを示します。

0x00

このテーブル エントリのデフォルトの動作を実行するようにエージェントに指定します。

例: システム上のすべてのプロセスの表示

以下の例では、ポート 1691 上で `sysedge` を実行しているローカル ホスト上のプロセスをすべて表示します（ダンプ）。

```
edgewatch -c private -p 1691 -v 1 -o process dump
```

プロセス モニタリング エントリの削除

プロセスのモニタリングを停止するには、プロセス モニタ テーブルおよび `sysedge.cf` ファイルから該当するエントリを削除する必要があります。CA Virtual Assurance ユーザ インターフェースからポイント設定を使用してエントリを削除すると、両方の場所からエントリを削除できます。手動の削除では、個別の操作で両方の場所からエントリを削除する必要があります。

CA Virtual Assurance 内でプロセス モニタ エントリを削除する方法

1. CA Virtual Assurance ユーザ インターフェースにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペイン内の [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバの詳細のページが表示されます。
4. [設定] をクリックし、[プロセス モニタ] をクリックします。
プロセス モニタ テーブルが、すべての既存のセルフ モニタと共に表示されます。
5. 削除するモニタを選択し、[アクション] - [削除] をクリックします。
テーブルからエントリが削除されます。エージェントは、変更を適用するためにウォーム スタートを実行します。

プロセス モニタ エントリを手動で削除する方法

1. ローカルの `sysedge.cf` ファイルからエントリの `watch process` ディレクティブを削除し、ファイルを保存します。
2. `-o` パラメータの以下のいずれかのコマンドを使用して、`edgewatch` ユーティリティを実行します。

```
-o delete index
```

```
-o setstatus index 6
```

```
index
```

削除するプロセス モニタ エントリのインデックス値を指定します。

`setstatus` コマンドを使用するときに値を 6 に指定すると、行ステータスが破棄されるように設定され、エントリが削除されます。

3. エージェントを再起動します。
エントリが削除されます。

Solaris ゾーン プロセスのモニタリング

フラグ `0x8000` (Solaris ゾーン プロセス) および変数 `pmonRegExpr` が適宜設定されている場合、SystemEDGE は Solaris ゾーン内のプロセスをモニタします。変数 `pmonRegExpr` は、形式が `ZoneRegExpr/ProcRegExpr` の正規表現を指定します。互換性の理由から、変数 `pmonZoneRegExpr` はまだ利用可能ですが、モニタが `0x8000` フラグが設定された `sysedge.cf` から初期化されていない場合のみ変更が許可されます。

変数 `pmonProcName` はすべてのプラットフォームで利用可能です。Solaris ゾーンがモニタされる場合、この変数の形式は `MatchedZoneName/MatchedProcessName` となります。Solaris ゾーンのモニタリングが必要でないか不可能な場合 (Solaris 10 OS 以外)、変数には `MatchedProcessName` のみが含まれます。

Solaris ゾーン内のプロセスをモニタするときは、フラグ `0x8000` が設定された `watch process` ディレクティブおよび形式が `ZoneRegExpr/ProcRegExpr` の正規表現を使用します。

アップグレード手順で、`watch zone` プロセス ディレクティブはすべて新しい形式に変換されます。

SystemEDGE はすべてのプロセス モニタ関連トラップ内の `pmonProcName` の値を送信します。

プロセスとサービスのモニタリングの推奨事項

プロセスとサービスのモニタリングを設定するときは、以下を検討します。

- 以下をモニタします。
 - プロセスのステータス
 - プロセスの状態を返すには、`processState`（オペレーティングシステムに依存する）または `processStateStr`（オペレーティングシステムに依存しない）OID を使用します。詳細については、[empire.asn1](#) を参照してください。
 - Perl などのインタープリタで処理されるプロセスの場合は、プロセスモニタリングエントリの作成時にプロセス名と引数（UNIX のみ）が一致するように `0x00000800` フラグを設定します。詳細については、「[プロセス モニタ テーブルのフラグ \(P. 305\)](#)」を参照してください。
 - 一定間隔の CPU 時間（この値が増分していない場合、プロセスはゾンビ状態である場合があります。）
 - 合計 CPU 時間（高い値は問題を示している可能性があります）
 - メモリ リーク（一定時間の RSS。問題をユーザに通知するためにアラームしきい値を使用します）
- 失敗したプロセスを自動的に再起動するように `SystemEDGE` を設定します。
- 競合状態を防ぐためにプロセスとサービスのモニタリングを無効にするように `UNIX` アプリケーションのシャットダウン ファイルを設定します。
- 同じ名前の複数のプロセスをモニタするには、プロセス グループ モニタ テーブルを使用します。詳細については、「プロセス グループ モニタリング」の章を参照してください。

第 10 章：自動ウォッチャー

このセクションには、以下のトピックが含まれています。

[自動ウォッチャーの仕組み](#) (P. 343)

[edgewatch ユーティリティ -- 自動ウォッチャーの使用](#) (P. 346)

[sysedge.cf での自動ウォッチャーの指定](#) (P. 357)

自動ウォッチャーの仕組み

自動ウォッチャーは、自動ウォッチャーがモニタを作成するリソースの名前に一致するパターンとして正規表現を使用して、定期的なディスカバリ プロセスを実行します。自動ウォッチャーを使用すると、新しいリソースがオンラインになるときに、これらに対するモニタを **SystemEDGE** で自動的に作成できます。自動ウォッチャーは予約済みのインデックス範囲（1000000 ～ 1999999）でモニタを作成します。

リソースが非表示になるとき、**SystemEDGE** は **CA Virtual Assurance** にトラップを送信し、自動ウォッチャーで設定した「消失アクション」を適用します。モニタ対象のリソースが消失する場合には、「消失アクション」によってモニタを削除したり、リソースのステータスを以下の特定のステータスに設定したりすることができます。

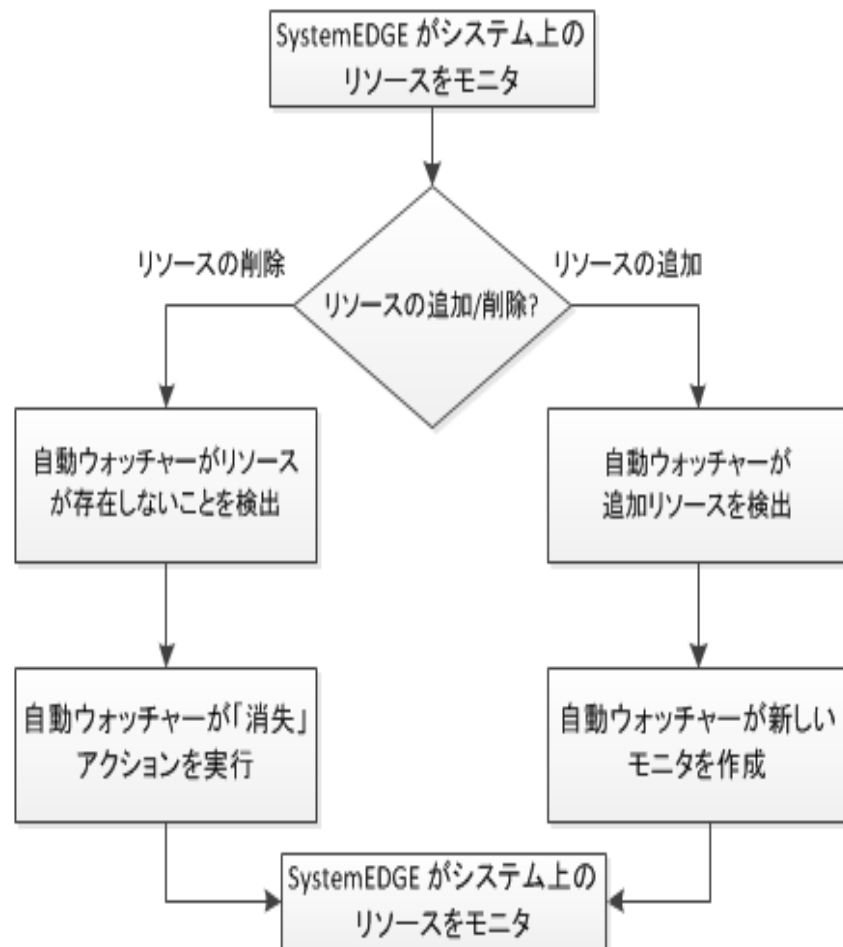
「OK」、「警告」、「マイナー」、「メジャー」、「重大」、「稼働中」、「ダウン」。

自動ウォッチャーを使用すると、管理対象システムに存在するリソースを知らなくても、柔軟にポリシーや階層型テンプレートを作成できます。リソースには、デバイス、サービス、管理対象システムで実行されるプロセスを指定できます。

使用できる自動ウォッチャーのタイプは以下のとおりです。

- 汎用自動ウォッチャー： デバイス、インターフェース、ファイルシステム、ファイルなど、管理対象システムのさまざまなリソース用のモニタを作成します。
- プロセスとサービスの自動ウォッチャー： 管理対象システムで実行されるプロセスとサービス用のモニタを作成します。

自動ウォッチャーのプロセス ワークフロー



消失アクションを設定するときには、以下のガイドラインを使用できます。

- 消失するリソースがシステムの健全性に影響する場合は、対応するリソースのステータスを重大な状態に変更する消失アクションを設定できます。
- 消失するリソースがシステムの健全性に影響しない場合は、対応するモニタを削除する消失アクションを設定できます。

関連項目:

[汎用自動ウォッチャー \(P. 345\)](#)

[プロセスとサービスの自動ウォッチャー \(P. 345\)](#)

汎用自動ウォッチャー

汎用自動ウォッチャーでは、デバイス、インターフェース、ファイルシステム、ファイルなど、管理対象システムのさまざまなリソース用のモニタを作成できます。

汎用自動ウォッチャーの例としては、以下のものがあります。

- 検出されたすべてのデバイスの容量
- 検出されたすべてのディスクのディスク サービス時間
- すべての `cmd` プロセスの常駐セット サイズ
- すべてのトンネル ネットワーク インターフェースの稼働ステータス
- すべてのデバイスのデバイス ステータス

関連項目:

[自動ウォッチャーの仕組み \(P. 343\)](#)

プロセスとサービスの自動ウォッチャー

プロセスとサービスのモニタを動的に作成するには、プロセスとサービスの自動ウォッチャーを使用します。

サービス自動ウォッチャーは、自動ウォッチャーの条件（サービス名、開始タイプなど）にサービスが一致した場合に、プロセス テーブルに複数のサービス モニタを作成します。たとえば、インストールされている `SQL` サービスで開始タイプが「自動」であるものをすべてモニタできます。

プロセス自動ウォッチャーは、以下の 2 つの方法でプロセス モニタを作成します。

- プロセス名を使用（デフォルト） - プロセス名が 自動ウォッチャーの条件に一致する場合。

たとえば、プロセス名が「sql」または「svchost」であるという条件にプロセスが一致するときに、プロセス モニタを作成します。 自動ウォッチャーによって作成されたプロセス モニタは、PID に関係なく、一致するプロセスで現在管理対象システムで実行されているものを追跡します。

- 自動ウォッチャーによって作成されたプロセス モニタの定義は、手動で作成されたプロセス モニタと同じです。
- 同じ名前で見数が異なるプロセスのセットを個別に管理できます。たとえば、「java.exe」などです。
- 関連するプロセスのセットに対するモニタを作成できます。

- PID の使用 - PID が 自動ウォッチャーの条件に一致する場合。 自動ウォッチャーでは、ユーザ インターフェースで PID フラグを使用してモニタ プロセスを有効にするか、または `sysedge.cf` ファイルで監視フラグ `0x1000` を指定します。

自動ウォッチャーによって作成された各モニタは、一致するプロセスのインスタンスをすべて追跡します。

- プロセスの特定のインスタンス用のモニタを作成します。
- ほかと区別するための引数を使用せず、複数のプロセスのインスタンスをモニタします。

関連項目：

[自動ウォッチャーの仕組み](#) (P. 343)

edgwatch ユーティリティ -- 自動ウォッチャーの使用

edgwatch コマンドラインユーティリティは、リソースのモニタに自動ウォッチャーを使用するように自動的に SystemEDGE エージェントを設定します。 特定の自動ウォッチャーおよび関連する引数を指定すると、edgwatch ユーティリティは `SNMP Set` リクエストを発行して、ターゲットエージェントの `MIB` テーブル内に該当するエントリを作成します。 edgwatch ユーティリティは、エージェントインストールの `bin` サブディレクトリにあります。

自動ウォッチャー用の edgwatch ユーティリティの使用方法は以下のとおりです。

```
edgwatch [-h hostname | ip_addr] [-p port] [-c community]
          [-v 1 | 2c | 3] [-u secName] [-s secLevel] [-n contextName]
          [-a authPassword] [-A MD5 | SHA]
          [-x privPassword] [-X DES | AES | 3DES]
          [-m FIPS_mode]
          [-r retries]
          [-t timeout] [-d logLevel] [-f logFile]
          -o autowatch command -attribute=value [-attribute=value ...]
```

-h hostname | ipaddr

エージェントが実行されているシステムのホスト名または IP アドレスを指定します。IPv4 アドレスおよび IPv6 アドレスを指定できます。

デフォルト: **localhost**

-p port

エージェントが実行されている UDP ポート（たとえば **1691**）を指定します。

デフォルト: **161**

-c community

エージェントが使用するコミュニティ文字列を指定します。**SNMPv1** および **SNMPv2c** の場合のみ有効です。

注: snmpset に使用する読み取り/書き込みコミュニティ文字列を指定します。

デフォルト: **public**

-v {1 | 2c | 3}

エージェントが実行している **SNMP** のバージョン。 **SNMPv1** の場合は **1** を、 **SNMPv2c** の場合は **2c** を、 **SNMPv3** の場合は **3** をそれぞれ指定します。

デフォルト： **1**

-u secName

SNMPv3 セキュリティに使用される **User-based Security Model (USM)** のユーザ名を指定します。

デフォルト： **none**

-s secLevel

以下に示した **SNMPv3** 通信用セキュリティ レベルのいずれかを指定します。

- 1 - noAuthNoPriv
- 2 - AuthNoPriv

3 - AuthPriv (SNMPv3 のみ)

-n contextName

MIBMuxed エージェントのインスタンス名を指定します。

デフォルト： **none**

-a authPassword

エージェントが **secLevel 2 (AuthNoPriv)** または **3 (AuthPriv)** の **SNMPv3** に設定されている場合に、認証パスワードを指定します。

注: **SNMPv3** 通信では、このオプションの指定は不要です。

デフォルト： なし

-A {MD5 | SHA}

エージェントが **secLevel 2 (AuthNoPriv)** または **3 (AuthPriv)** の **SNMPv3** に設定されている場合に使用する認証プロトコルを指定します。現在、**MD5 (Message Digest Algorithm)** および **SHA (Secure Hash Algorithm)** のみを使用できます。

デフォルト: **MD5**

-x privPassword

エージェントが **secLevel 3 (AuthPriv)** の **SNMPv3** に設定されている場合に、プライバシー (暗号化) パスワードを指定します。

デフォルト: なし

-X {DES | AES | 3DES}

SNMPv3 ユーザが **secLevel 3 (AuthPriv)** で設定されている場合に、プライバシー プロトコルを指定します。データ暗号化標準の場合は **DES**、**128** ビットの暗号化キーを使用した高度暗号化標準 (**AES128**) の場合は **AES**、トリプルデータ暗号化標準の場合は **3DES** と指定します。

デフォルト: なし

-m FIPS_mode

操作の **FIPS** モードを制御します。許可される値は **0**、**1**、および **2** です。

0

非 **FIPS** モード

1

FIPS 共存モード

2

FIPS のみのモード

デフォルト: **1**

-r retries

再試行の回数を指定します。

デフォルト: **10**

-t timeout

SNMP レシーバが要求をタイムアウトしたと判断するまでの期間を指定します。

デフォルト : **10 秒**

-d logLevel

SNMP メッセージのログ レベルを指定します。許可される値は **0 ~ 5** です。

0

致命的なメッセージをログ記録します。

1

重大なメッセージをログ記録します。

2

警告メッセージをログ記録します。

3

情報メッセージをログ記録します。

4

すべてのメッセージをログ記録します。

5

デバッグ メッセージを含むすべてのメッセージをログ記録します。

デフォルト : **0**

-f logfile

エラーとデバッグ情報が含まれるログ ファイルの名前を指定します。

デフォルト : **sysedge_utility.log**

コマンド

コマンドおよび関連する属性を指定します。サポートされるコマンドは以下のとおりです。

- `add -index=[watchIndex] [additional attributes ... ]`
- `setstatus -index=[watchIndex] -rowstatus=[active | notInService | destroy]`
- `delete -index=[watchIndex]`
- `list`

注: コマンドの詳細については、「[自動ウォッチャー用の edgwatch コマンド](#) (P. 351)」を参照してください。

自動ウォッチャー用の edgwatch コマンド

`edgwatch autowatch` コマンドおよび関連する引数は以下のとおりです。

```
add -index=[watchIndex] -attribute=value [-attribute=value ...]
setstatus -index=[watchIndex] -rowstatus=[active | notInService | destroy]
delete -index=[watchIndex]
list
```

属性はすべて「`-attribute=value`」形式で指定し、位置に依存しません。

`add` コマンドの必須属性は以下のとおりです。

`-name`

自動ウォッチャー エントリの名前を指定します。

`-watchtype`

以下のいずれかのタイプを指定します。**generic**、**process**、または **service**。

`-criteria`

モニタ対象インスタンスを選択するためのプライマリ テーブル条件の正規表現を単一引用符で囲んで指定します。

`-severity`

オブジェクト状態モデルに使用する次のいずれかの重大度を指定します：**none**、**ok**、**warning**、**minor**、**major**、**critical**、**fatal**。

add -watchtype=generic の追加必須属性は以下のとおりです。

-table

モニタ対象のテーブルエントリを指定します (**devTableEntry** など)。

-attribute

モニタ対象のテーブル列属性を指定します (**devCapacity** など)。

-op

次の比較演算子を指定します: **nop**、**gt**、**lt**、**ge**、**le**、**eq**、**ne**。

-value

比較のしきい値を指定します。

add -watchtype=process の追加必須属性は以下のとおりです。

-attribute

次のいずれかの値を指定します: **procAlive**、**procMem**、**procSize**、**procRSS**、**procTime**、**procInBlks**、**procOutBlks**、**procMsgsSent**、**procMsgsRecv**、**procNice**、**procNumThreads**、**procNumSwaps**、**procSysCalls**、**procMinorPgFlts**、**procMajorPgFlts**、**procVolCtx**、**procInvolCtx**、**procTimePermil**。

-op

次の比較演算子を指定します: **nop**、**gt**、**lt**、**ge**、**le**、**eq**、**ne**。 **procAlive** には必要ありません。

-value

比較のしきい値を指定します。

オプション属性

-interval

生成されるモニタの評価間隔を指定します (デフォルト: **60**)。

-desc

生成されるすべてのモニタに適用される説明を単一引用符で囲んで指定します。

-monflags

生成されるすべてのモニタに適用される 16 進数の整数 (0x0000 など) を指定します (デフォルト: 0x0000)。

0x0001 - アクションを実行しない。

0x0002 - トラップを送信しない。

0x0004 - エントリを再初期化しない。

0x0008 - syslog イベントを送信しない。

0x0010 - 連続する notReady トラップを送信する。

0x0020 - アクション スクリプトに SystemEDGE 引数を送信しない。

0x0040 - このエントリの notReady トラップを送信しない。

-watchtype = generic :

0x0100 - クリア トラップを送信する。

0x0200 - X 回の違反後にしきい値トラップを送信する。

0x0400 - 最大 X 回の連続するしきい値トラップを送信する。0xffff0000 = X の値。

-watchtype = process または service :

0x0200 - 開始トラップを送信しない。

0x0400 - クリア トラップを送信する。

0x0800 - プロセス名および引数を一致させる。

0x1000 - X 回の違反後にしきい値/停止トラップを送信する。

0x2000 - 最大 X 回の連続するしきい値/停止トラップを送信する。

0x8000 - Solaris ゾーンでプロセスをモニタする。

0xffff0000 - X の値。

-matchtype

プライマリ選択条件として使用する以下のいずれかの一致タイプを指定します: **positive**、**negative** (デフォルト: **positive**)。

-criteria2

セカンダリ選択条件として使用するモニタ対象テーブル内の列を指定します。

-criteria2regex

セカンダリ選択条件で使用する正規表現を単一引用符で囲んで指定します。（デフォルト：'!.*'）

-matchtype2

セカンダリ選択条件として使用する以下のいずれかの一致タイプを指定します：**positive**、**negative**（デフォルト：**positive**）。

-criteria3

ターシャリ選択条件として使用するモニタ対象テーブル内の列を指定します。

-criteria3regex

ターシャリ選択条件で使用する正規表現を単一引用符で囲んで指定します。（デフォルト：'!.*'）

-matchtype3

ターシャリ選択条件として使用する以下のいずれかの一致タイプを指定します：**positive**、**negative**（デフォルト：**positive**）。

-stype

生成されるすべてのモニタに適用される以下のいずれかのサンプルタイプを指定します：**absolute**、**delta**（デフォルト：**absolute**）。

-scale

生成されるすべてのモニタに適用されるスケールの値を指定します（デフォルト：**1**）。

-action

コマンド（**action**）のフルパスを単一引用符で囲んで指定します（デフォルト：**action** は未設定）。

-objclass

オブジェクト状態モデルのオブジェクト クラスの名前を指定します。

-objattr

オブジェクト状態モデルのオブジェクト属性の名前を指定します。

-watchflags

自動ウォッチャーの動作を変更するための 16 進数の整数 (0x0000 など) を指定します (デフォルト: 0x0000)。

0x0002 - 自動ウォッチャー トラップを送信しない。

0x0004 - エントリを再初期化しない。

0x0008 - syslog イベントを送信しない。

0x0010 - 連続する notReady 自動ウォッチャー トラップを送信する。

0x0040 - このエントリの notReady トラップを送信しない。

0x1000 - PID でプロセスをモニタする (-watchtype=process の場合のみ)。

-scaninterval

モニタ対象テーブルのディスカバリ スキャンの間隔を指定します (デフォルト: 300)。

-limit

この自動ウォッチャー エントリ用に生成されるモニタの最大数を指定します (デフォルト: 100)。

-loss

生成されたモニタがそのオブジェクトをモニタしなくなったときに実行される次のいずれかの消失アクションを指定します: **remove**、**ok**、**warning**、**minor**、**major**、**critical**、**fatal**、**up**、**down** (デフォルト: **fatal**)。

-starttype

Windows サービスの開始タイプを指定します。この属性は **-watchtype=service** にのみ適用されます。次のいずれかの値を使用します: **automatic**、**manual**、**disabled** (デフォルト: すべてのタイプ)。

setstatus

setstatus 操作と共に使用されたときに MIB テーブルで行のステータスを設定する際に使用する、**Status** テキスト表記法の値を指定します。有効な値は以下のとおりです。値は、割り当てられた整数値またはスペルアウトされた実際のステータス テキストのいずれかになります。

- active(1)
- notInService(2)
- destroy(6)

edgework 自動ウォッチャーの例

このセクションでは、自動ウォッチャーを作成するために、SNMP バージョン 1、2c、および 3 で edgework ユーティリティを使用する例を示します。

例:

汎用自動ウォッチャー エントリを (テーブル インデックス 13 に) 追加して、しきい値が 90 の検出されたすべてのファイルシステムを対象とする devCapacity モニタを生成します。

```
edgework -c private -o autowatch add -index=13 -watchtype=generic
        -name='All FileSystems - Minor' -table=devTableEntry
        -attribute=devCapacity -interval=120 -criteria='.*' -op=ge
        -value=95 -severity=minor -desc='FileSystem Usage (Percentage)'
        -objclass=FileSystem -objattr=PercentUsed
```

プロセス自動ウォッチャー エントリを (テーブル インデックス 15 に) 追加して、検出されたすべての SQL プロセスを対象とする procRSS モニタを生成し、プロセスの終了時にこれらのモニタを自動的に削除します。

```
edgework -c private -o autowatch add -index=15 -watchtype=process
        -name='SQL Processes RSS' -criteria='.*sql.*'
        -attribute=procRSS -op=ge -value=100000 -severity=major
        -interval=60 -desc='SQL processes' -objclass='Process'
        -objattr='ResidentSet' -loss=remove
```

サービス自動ウォッチャー エントリを (テーブル インデックス 17 に) 追加して、自動開始用に設定されたすべてのサービスを対象とするサービス procAlive ウォッチャーを生成します。

```
edgework -c private -o autowatch add -index 17 -watchtype=service
        -name='All Automatic Services' -criteria='.*' -severity=major
        -interval=300 -limit=200 -desc='Automatic Services'
        -objclass='Service' -objattr='Running' -starttype=automatic
```

定義されているすべての自動ウォッチャー エントリを一覧表示します。

```
edgework -c private -o autowatch list
```

以前に指定されたサービス自動ウォッチャー エントリを削除します。

```
edgework -c private -o autowatch delete -index=17
```

以前に指定されたプロセス自動ウォッチャーの行ステータスを notInService に設定します。

```
edgework -c private -o autowatch setstatus -index=15 -rowstatus=notInService
```

sysedge.cf での自動ウォッチャーの指定

自動ウォッチャーは、指定された自動ウォッチャーの条件に一致するリソース用のモニタおよびプロセスモニタ **MIB** テーブルにエントリを作成します。自動ウォッチャーディレクティブは、自動的に生成されるモニタの条件を指定します。

自動ウォッチャーの構文は、スイッチに基づき、パラメータの位置に依存しません。すべてのパラメータの形式は以下のとおりです。

```
-parameter=value  
-parameter='value'
```

オプションのパラメータは省略できます。値にスペースが含まれる場合は、単一引用符を使用します。

自動ウォッチャーディレクティブの形式は、以下のとおりです。

```
autowatcher -index=index -watchtype=type -table=table -attribute=attr ...
```

必須パラメータ:

-index

自動ウォッチャーを格納するために使用されるテーブルインデックスを指定します。

-name

自動ウォッチャーの名前を指定します。

-watchtype

作成する自動ウォッチャーのタイプ (**generic**、**service**、**process**) を指定します。

-table

モニタ対象のテーブルエントリを指定します (**devTableEntry** など)。サービスまたはプロセス自動ウォッチャーには必要ありません。

-attribute

モニタする属性を指定します (**devCapacity** など)。サービス自動ウォッチャーには必要ありません。

-criteria

モニタ対象インスタンスを選択するためのプライマリ条件の正規表現を指定します。

-op

比較演算子（**nop**、**gt**、**lt**、**ge**、**le**、**eq**、**ne**）を指定します。

-value

しきい値を指定します（プロセス **procAlive**、またはサービス自動ウォッチャーには必要ありません）。

-severity

オブジェクト状態モデルに使用する重大度を指定します（**none**、**ok**、**warning**、**minor**、**major**、**critical**、**fatal**）。

オプション パラメータ:

-interval

この自動ウォッチャーが生成するすべてのモニタの評価間隔を指定します。

デフォルト : **60**

-desc

この自動ウォッチャーが生成するすべてのモニタに適用される説明を指定します。

-monflags

この自動ウォッチャーが生成するすべてのモニタに適用されるセルフモニタ フラグのセットを指定します。

-matchtype

プライマリ選択条件として使用する一致タイプを指定します（**positive** または **negative**）。

-criteria2

セカンダリ選択条件として使用するモニタ対象テーブル内の列を指定します。

-criteria2regex

セカンダリ インスタンス選択条件で使用する正規表現を指定します。

-matchtype2

セカンダリ選択条件として使用する一致タイプを指定します (**positive** または **negative**)。

-criteria3

ターシャリ選択条件として使用するモニタ対象テーブル内の列を指定します。

-criteria3regex

セカンダリ インスタンス選択条件で使用する正規表現を指定します。

-matchtype3

セカンダリ選択条件として使用する一致タイプを指定します (**positive** または **negative**)。

-stype

各モニタに適用されるサンプル タイプを指定します (**absolute**、**delta**)。

-scale

収集されたモニタ値のスケール係数を指定します。スケール係数を使用すると、変数が最初に測定されたさまざまなスケール単位で変数をモニタできます。

スケール係数 **1024** を使用して、元はバイトで測定されていた変数の値をキロバイトでモニタします。

このスケール係数は、変数 **monCurrVal**、**monVal**、**monMinValue**、および **monMaxValue** を、バイトではなく、キロバイトでモニタします。

デフォルト : **1**

-action

各モニタのトラップの送信と同時に実行されるコマンドのフルパス (パラメータを含む) を指定します。

-objclass

オブジェクト状態モデルに使用するオブジェクト クラスを指定します (" はデフォルト値に置き換えられます。「*」は無効です)。

-objattr

オブジェクト状態モデルに使用するオブジェクト属性を指定します (" はデフォルト値に置き換えられます。「*」は無効です)。

-watchflags

自動ウォッチャーの動作を変更するフラグを指定します。

-scaninterval

モニタ対象テーブルのスキャンの間隔を秒単位で指定します。

-limit

この自動ウォッチャーが生成できるモニタの最大数を指定します。

-loss

モニタがモニタ対象オブジェクトに到達できなくなったときに実行される消失アクションを指定します(**remove**、**ok**、**warning**、**minor**、**major**、**critical**、**fatal**、**up**、**down**)。

すべての自動ウォッチャーに有効なフラグ

000 000 02 - トラップを送信しないが、トラップをログ記録する。

000 000 04 - **notReady** 自動ウォッチャーを再初期化しない。

000 000 08 - トラップをログ記録しないが、トラップを送信する。

000 000 10 - **notReady** トラップの送信を続ける。

000 000 40 - **notReady** トラップ (ログを含む) を送信しない。

000 000 80 - **AIM** プラグインによってこのエントリが作成されている。
このフラグは読み取り専用です。

関連項目:

[汎用自動ウォッチャーの例](#) (P. 361)

[サービス自動ウォッチャーの例](#) (P. 362)

[プロセス自動ウォッチャーの例](#) (P. 362)

汎用自動ウォッチャーの例

以下の例では、sysedge.cf ファイルの汎用自動ウォッチャーを指定します。

- 検出されたすべてのデバイスで容量をモニタします。
- 検出されたすべてのディスクでディスク サービス時間をモニタします。
- すべての cmd プロセスで常駐セット サイズをモニタします。
- すべてのトンネル ネットワーク インターフェースの稼働ステータスをモニタします。
- すべてのデバイスのデバイス ステータスをモニタします。

```
# Device Table - Monitor capacity on all discovered devices
# (minor at 50%, major at 75%)
autowatcher -watchtype=generic -index=11 -table=devTableEntry
-name='Device Table Autowatcher' -criteria='.*' -attribute=devCapacity -op=ge
-value=50 -severity=minor -desc='Device capacity (percentage)' -monflags=0x0
-limit=10 -interval=60 -objclass='Disk' -objattr='Capacity'
autowatcher -watchtype=generic -index=12 -table=devTableEntry
-name='Device Table Autowatcher' -criteria='.*' -attribute=devCapacity -op=ge
-value=75 -severity=major -desc='Device capacity (percentage)' -monflags=0x0
-limit=10 -interval=60 -objclass='Disk' -objattr='Capacity'
# Disk Statistics Table - Monitor disk service time on all discovered disks
autowatcher -watchtype=generic -index=101 -table=diskStatsEntry
-name='Disk Stats Autowatcher' -criteria='.*' -attribute=diskStatsServiceTime
-op=ge -value=100 -severity=major -desc='Disk service time' -interval=60
-objclass='DiskStats' -objattr='ServiceTime'
# Process Table - Monitor the resident set size on all cmd processes
# lossAction example
autowatcher -watchtype=generic -index=202 -table=processEntry
-name='Process Table Autowatcher' -criteria='cmd' -attribute=processRSS
-op=ge -value=10000 -severity=major -desc='cmd process RSS (KB)' -interval=30
-objclass='Process' -objattr='process RSS' -scaninterval=30
# MIB-2 Interface Table - Monitor the operating status of all tunneled network
# interfaces - multiple criteria example
autowatcher -watchtype=generic -index=303 -table=ifEntry
-name='WiFi Interface Autowatcher' -criteria='.*' -criteria2=ifType
-criteria2regex='131' -attribute=ifOperStatus -op=eq -value=2
-desc='tunnel interface operating status' -severity=critical -objclass=Network
-objattr=OperatingStatus
# Host Resource Device Table - Monitor the device status of all devices
# dual-column instance example
autowatcher -watchtype=generic -index=404 -table=hrDeviceEntry
-name='HR Device Autowatcher' -criteria='.*' -attribute=hrDeviceStatus
-op=ne -value=2 -desc='HR device status' -severity=fatal -objclass=Hardware
-objattr=OperatingStatus
```

サービス自動ウォッチャーの例

以下の例では、sysedge.cf ファイルのサービス自動ウォッチャーを指定します。自動ウォッチャーは、指定された条件（サービス名、開始タイプなど）に一致する複数のサービス モニタをプロセス テーブルに作成します。

- 開始タイプが「automatic」のインストールされたすべての SQL サービスをモニタします。

```
autowatch -index=12 -watchtype=service -name='All Automatic SQL Services'
-criteria='.*SQL.*' -starttype=automatic -interval=60 -severity=critical
-desc='SQL services' -objclass='Service' -objattr='Running'
```

プロセス自動ウォッチャーの例

以下の例では、sysedge.cf ファイルのプロセス自動ウォッチャーを指定します。

- eTrust プロセスはどれもプロセッサ使用率が 5% を超えていないことを確認します。

```
autowatch -index=22 -watchtype=process -name='eTrust Processes'
-criteria='^Ino.*' -attribute=procTimePermil -op=ge -value=50000 -severity=major
-interval=60 -desc='eTrust processes' -objclass='Process' -objattr='CPU_Usage'
```

- すべての SVCHOST プロセスが使用する実メモリが 10 MB 未満であることを確認します。

```
autowatch -index=23 -watchtype=process -name='Svchost Processes'
-criteria='^svchost' -attribute=procRSS -op=ge -value=10000 -severity=major
-interval=60 -monflags=0x0800 -desc='Svchost processes' -objclass='Process'
-objattr='ResidentSet'
```

第 11 章: プロセス グループ モニタリング

この章では、SystemEDGE エージェントを使用してグループのプロセスをモニタする方法について説明します。エージェントは、プロセス グループを使用して、プロセスごとの情報をポーリングおよびモニタしやすい 1 つの値に集約します。

このセクションには、以下のトピックが含まれています。

[プロセス グループ モニタリングの概要](#) (P. 363)

[\[プロセス グループ モニタ\] テーブル](#) (P. 364)

[プロセス グループ モニタリングの設定](#) (P. 377)

[edgemonitor ユーティリティ -- プロセス グループのモニタ](#) (P. 384)

[プロセス グループ モニタリング エントリの削除](#) (P. 390)

[Solaris ゾーン プロセス グループ モニタリング](#) (P. 391)

プロセス グループ モニタリングの概要

SystemEDGE エージェントでは、基礎となるシステムで実行されているプロセスのグループを動的にモニタできます。[プロセス グループ モニタ] テーブルを使用して、プロセス グループ、正規表現、間隔、および重大度を指定します。また、エージェントは、その情報を使用して、正規表現に一致するプロセスのグループをモニタします。

SystemEDGE エージェントは、プロセス グループをモニタして、各グループ内に存在するプロセスと、グループ メンバシップが変化するかどうかを判断します。アプリケーションのコンポーネントが開始または失敗した場合や、メンバがグループから削除された場合やグループに追加された場合、SystemEDGE エージェントはトラップにういて管理システムに自動的に通知できます。

エージェントは、[プロセス グループ モニタ] テーブルの読み取り専用列に、プロセス グループの組み合わせられたメトリック値も入力します。セルフ モニタリング エントリを作成して、定義済みのしきい値に基づいたこれらの属性をモニタできます。

注: 個々のプロセスと Windows サービスのモニタの詳細については、「プロセスとサービスのモニタリング」の章を参照してください。

[プロセス グループ モニタ]テーブル

[プロセス グループ モニタ] テーブルには、エージェントが現在モニタしているプロセスの各グループに関する情報が存在します。各行は、定義されたプロセスグループ モニタ エントリを表しています。

テーブルには、エントリごとに以下のタイプの情報が存在します。

- グループ内のプロセスのリスト
- エージェントがグループを確認する間隔
- グループ合計を使用したプロセス属性の値
- モニタ重大度

[プロセス グループ モニタ]テーブルの列

以下のリストでは、[プロセス グループ モニタ] テーブルの列について説明します。[プロセス グループ モニタ] テーブルおよびそのフィールドの詳細な説明については、Systems Management Empire MIB の仕様（エージェント インストールの mib サブディレクトリ内の empire.asn1）を参照してください。

注: 個々のプロセスがグループから削除された後も、SystemEDGE にはグループ メンバシップの履歴が保持され、プロセス統計が追跡されます。

pgmonIndex

このエントリの一意の行インデックス（1 ～ MAXINT）を定義します。行 1 ～ 10 はエージェントの内部使用のために予約されています。追加の行のインデックスは 11 ～ MAXINT の範囲にする必要があります。

権限：読み取り専用

pgmonDescr

エントリの説明（0 ～ 512 文字の長さ）を定義します。

権限：読み書き

pgmonInterval

エージェントがグループをモニタする頻度を秒単位で定義します。値は 30 秒以上かつ 30 秒の倍数にする必要があります。

権限：読み書き

デフォルト：60

pgmonProcRegExpr

エージェントが名前でプロセスを一致させようとするときに使用する正規表現を定義します。

権限：読み書き

pgmonFlags

エントリがその操作の過程で従う追加の動作セマンティクスを示す正の整数フラグを定義します。利用可能なフラグおよびフラグの設定の詳細については、「プロセス (372P.)グループ モニタ」テーブルのフラグ」を参照してください。

権限：読み書き

pgmonNumProcs

エントリが追跡しているプロセス グループ内のプロセスの現在の数を定義します。その名前（場合によっては、その引数）がこのエントリに設定された正規表現に一致する場合、プロセスはそのプロセス グループに属しています。

権限：読み取り専用

pgmonPIDList

プロセス グループに数値 **PID** を定義します。各 **PID** はスペース文字によって次の **PID** と区切られます。

権限：読み取り専用

pgmonStatusList

プロセス グループに各プロセスのプロセス ステータスを定義します。各プロセス状態はスペース文字で次の状態と区切られます。ステータス リスト内のエントリは、**PID** リストのエントリと 1 対 1 で対応しています。プロセスの状態の詳細については、**processStateStr** MIB 変数を参照してください。

権限：読み取り専用

pgmonAction

式が **True** と評価され、エージェントがプロセス グループ変更トラップを送信したときに実行する、パラメータ付きのコマンド（0 ～ 2048 文字の長さ）を引用符付きで定義します。文字列が空の場合、エージェントはこのエントリのアクションを呼び出しません。

注:アクションに **Windows** バッチ ファイルを使用しないでください。プログラム上の厳しい制限があり、デスクトップアプリケーションでは正しく動作しないことがよくあります。代わりに、**Perl** または **Visual Basic** など、強力な柔軟なスクリプト言語を使用してください。

権限：読み書き

pgmonNumEvents

エントリに生成されたイベントの数を定義します。イベントは必ずしもトラップを意味するわけではありません。トラップは、フラグ設定をとおして行で無効になっている場合があります。

権限：読み取り専用

pgmonNumTraps

エージェントがこのエントリに送信したトラップ
(**procGroupChangeTrap**) の数を定義します。

権限：読み取り専用

pgmonLastTrap

エージェントがこのエントリに最後にトラップ
(**procGroupChangeTrap**) を送信した時間 (**sysUpTime** に基づく) を定義します。

権限：読み取り専用

pgmonRowStatus

行ステータスを定義します。以下のいずれかになります。

- active
- notInService
- notReady
- createAndGo
- createAndWait

通常、行は **active** または **notInService** のいずれかです。これらの値は、**SNMPv2** の **SMI RowStatus** テキスト表記法によって定義された値と意味が同一です。

権限：読み書き

デフォルト：**createAndWait(5)**

pgmonRSS

プロセスのグループの組み合わせられた常駐セットサイズ (**RSS**) を定義します。グループ内の各プロセスの **RSS** は間隔ごとに合計され、この変数に格納されます。**RSS** には共有メモリも含まれるので、グループの **RSS** の合計は基礎となるシステムで使用可能な物理メモリの合計を超える可能性があります。詳細については、システム管理 **MIB** および **Systems Management Empire MIB** の **processRSS** 変数を参照してください。権限：読み取り専用

pgmonSize

プロセスのグループのテキスト、データ、およびスタック セグメントの組み合わせられたサイズを定義します。グループ内の各プロセスのサイズは間隔ごとに合計され、この変数に格納されます。サイズには共有メモリが含まれるので、グループのサイズの合計は基礎となるシステムの仮想メモリの合計を超える可能性があります。詳細については、システム管理 MIB および Systems Management Empire MIB の **processSize** 変数を参照してください。権限：読み取り専用

pgmonThreadCount

プロセスのグループのスレッドの総数を定義します。グループの各プロセスで実行されるスレッドの数は、間隔ごとに合計され、この MIB 変数に格納されます。詳細については、Systems Management Empire MIB (empire.asn1 内) の **processNumThreads** 変数を参照してください。

権限：読み取り専用

pgmonMEM

このグループのプロセスによって使用されている実メモリの割合の合計を定義します。使用されているメモリの割合は間隔ごとに合計され、この MIB 変数に格納されます。メモリ使用量には共有メモリ（共有ライブラリおよび DLL）が含まれるので、割合の合計は 100 を超える可能性があります。

権限：読み取り専用

pgmonInBlks

このグループのプロセスによって入力されるデータのブロックの数を定義します。

権限：読み取り専用

pgmonOutBlks

このグループのプロセスによって出力されるデータのブロックの数を定義します。

権限：読み取り専用

pgmonMsgsSent

このグループのプロセスによって送信されるメッセージの数を定義します。

権限：読み取り専用

pgmonMsgsRecv

このグループのプロセスによって受信されたメッセージの数を定義します。

権限：読み取り専用

pgmonSysCalls

このグループのプロセスによって呼び出されたシステム コールの数を定義します。

権限：読み取り専用

pgmonMinorPgFlts

このグループのプロセスに生じたマイナー ページフォールトの数を定義します。

権限：読み取り専用

pgmonMajorPgFlts

このグループのプロセスに生じたメジャー ページフォールトの数を定義します。

権限：読み取り専用

pgmonNumSwaps

このグループのプロセスがスワップされた回数を定義します。

権限：読み取り専用

pgmonVolCtx

このグループのプロセスに生じた任意コンテキスト切り替えの数を定義します。

権限：読み取り専用

pgmonInvolCtx

このグループのプロセスに生じた強制コンテキスト切り替えの数を定義します。

権限：読み取り専用

pgmonCPUSecs

このグループのプロセスによって使用される CPU 時間の秒数を定義します。

権限：読み取り専用

pgmonMatchUser

ユーザ名と、有効なユーザ名に設定されているときは任意のプロセス名正規表現が同じ、実行中のプロセスに一致します。この変数は **UNIX** システムでのみ有効です。

権限：読み書き

pgmonMatchGroup

グループ名と、プロセス名正規表現、有効なグループ名に設定されているときはユーザ名が同じ、実行中のプロセスに一致します。この変数は **UNIX** システムでのみ有効です。

権限：読み書き

pgmonProcNameList

モニタされたプロセスのプロセス名のリスト。 **Solaris 10** 以降では、ゾーン正規表現が使用されている場合、名前の形式は **ZoneName/ProcName** です。

権限：読み取り専用

行作成オブジェクト

[プロセス グループ モニタ] テーブルの **pgmonIndex** 列は、テーブル内の行を識別するためにキー フィールド（または行インデックス）として機能します。行 1 ～ 10 は、SystemEDGE エージェントによる内部使用のために予約されています。11 ～ MAXINT の範囲の行を設定できます。

行作成を最適化するには、[プロセス グループ モニタ] テーブルと共に以下の MIB オブジェクトを使用できます。

pgmonUnusedIndex

オブジェクトで **SNMP Get** を実行するとき、[プロセス グループ モニタ] テーブルの未使用のインデックス番号を返します。このオブジェクトをクエリして、行作成に使用できる番号を取得します。

pgmonMatchDescr

pgmonDescr 属性の特定のエントリ説明に相当するインデックス番号を決定します。**pgmonDescr** 属性値を使用してこの MIB オブジェクト上で **SNMP Set** を実行して、エージェントが [プロセス グループ モニタ] テーブル内のエントリを検索し、最後に一致したエントリのインデックス値を **monMatchIndex** MIB オブジェクトに配置するようにします。

pgmonMatchIndex

pgmonMatchDescr と共に使用すると、インデックス番号を持つ特定のエントリ説明に一致します。

ローカル ポリシーに基づいて、行のブロックをシステム管理者に予約するように選択することができます。その場合、行がすでに別のユーザのエントリによって取得されていることを心配しなくても、行の予約済みブロック内のエントリを定義できます。他のすべてのユーザがユーザ設定のエントリを定義する場合は、ローカル ポリシーに従って、予約された範囲外にある行インデックスを使用する必要があります。

行のブロックを予約することによって、一貫した条件のセット（行エントリ）をすべてのコンピュータにおいてモニタできるよう定義することが可能となり、各コンピュータの同じ行番号に同じ条件が定義されるようになります。たとえば、行 11（monIndex=11）を使用して、**swapCapacity** 変数をモニタするためのエントリを定義でき、すべてのシステムで行 11 が **swapCapacity** 変数をモニタするように、すべてのシステムにこの設定を配布できます。

CA Virtual Assurance でエージェントを設定すると、このプロセスを自動化できます。特定の行番号を持つエントリのブロックを設定し、1 回の操作でのすべての管理対象システムにその設定を展開できます。また、CA Virtual Assurance は、すでに使用中の行が定義されているかどうかを検出し、操作を禁止します。

CA Virtual Assurance での行の定義の詳細については、CA Virtual Assurance のドキュメントを参照してください。

プロセス グループ モニタ テーブルのフラグ

プロセス グループ モニタ テーブル内の **pgmonFlags** 列は、対応する行に追加の動作セマンティックスを指定できる 32 ビット符号なし整数フィールドです。

デフォルトでは、プロセス グループ モニタ テーブル行は以下を実行します。

- 自分を再初期化しようとする
- SNMP トラップを送信する
- 共通ログ ファイルにイベントを記録する
- アクションを呼び出す（アクションが設定されている場合）

これらのデフォルトを変更するように、さまざまなフラグ ビットを設定できます。このエージェントは、16 進（基数 16）法で表記されたすべてのフラグを解釈します。

プロセス グループ モニタ テーブルのフラグ (pgmonFlags) を以下に示します。

フラグ値は 3 つのフィールドから構成されます。

- フィールド 1 には、Systems Management Empire MIB のモニタ テーブル用に定義された共通のテーブルフラグが含まれます。この部分はフラグの低位 8 ビットです。
- フィールド 2 には、各モニタ テーブル用に個別に定義された、テーブル固有のフラグが含まれます。このフィールドは、共通テーブルフラグの次の低位 12 ビットを定義します。
- フィールド 3 には、テーブル固有のフラグで使用する整数値用の予約済み高位 12 ビットが含まれます。このフィールドは、プロセス グループ モニタ テーブルに固有のフラグを定義します。

以下のセクションでは、各フラグ ビットについて説明します。OR 論理演算によってフラグの値を組み合わせることができます。1 つのフラグ (0x00100) はプロセス グループ モニタ テーブルに固有です。このフラグを設定すると、SystemEDGE エージェントは、このエントリのプロセス名および引数に一致させます。

以下のリストではプロセス モニタ テーブルのフラグについて説明します。

0x00000001

このエントリに対して、アクションが実行されないようにします。

0x00000002

このエントリに対して、SNMP トラップが送信されないようにします。

0x00000004

このエントリの再初期化を無効にします。デフォルトでは、エージェントは、ターゲットプロセスが再起動されている場合に新しいプロセス ID を確認するためにプロセス テーブルをスキャンすることにより、定期的に、このエントリを再初期化しようとします。

0x00000008

sysedge.log ファイル内で、このエントリに対し、トラップのログが記録されないようにします。このビットを設定しても、トラップの送信には影響しません。イベントが頻繁に発生する場合、または特定のエントリをエージェントのハートビートとして使用している場合は、イベントのログ記録を無効にすると有用です。

0x00000020

アクションスクリプトまたはプログラムへのデフォルト引数の受け渡しを無効にします。**SystemEDGE** は、通常、トラップタイプ、説明フィールドなどを示すデフォルト アクション パラメータを渡します。アクション パラメータの詳細については、「[プロセス グループ モニタ テーブルのアクション パラメータ \(P. 374\)](#)」を参照してください。

0x00000040

このエントリに対して **notReady** トラップが送信されないようにします。**notReady** トラップのログは記録されず、**notReady** トラップに対してアクションも実行されません。

0x00000100

このエントリのプロセス名と引数に一致させます。

0x00008000

Solaris ゾーンのプロセスに一致させます。**pgmonProcRegExpr** 属性は形式 **ZoneRegExpr/ProcRegExpr** であることが想定されます。

[プロセス グループ モニタ]テーブルのアクション パラメータ

アクション コマンドが呼び出されると、**SystemEDGE** エージェントはそれらのアクション コマンドにいくつかのデフォルト パラメータを渡します。これらのパラメータは、アクション文字列で指定したパラメータに加えて使用され、指定したパラメータの後コマンドラインに渡されます。デフォルト パラメータは、[プロセス グループ モニタ]テーブルに送信される **SNMP** トラップで使用するパラメータと同じです。

注: テーブル エントリで **0x00000020** フラグを設定することによって、エージェントがこれらのアクション パラメータをコマンドに含めないようにすることができます。

以下のリストでは、[プロセス グループ モニタ] テーブル アクションのデフォルト パラメータについて説明します。

pgmonIndex

テーブル エントリのインデックスを定義します。

pgmonDescr

テーブル エントリの説明を定義します。

pgmonFlags

テーブル エントリと関連付けられたフラグを定義します。アクション スクリプトに渡される値は、**16 進数**（たとえば **0x00000020** など）であることを示すために、先頭が **0x** の **16 進表記**になっています。

pgmonNumProcs

エントリがモニタするプロセスの数を定義します。

pgmonRegExpr

グループとしてモニタするプロセスのプロセス ID を検索するために エージェントが使用する正規表現を定義します。

pgmonRowStatus

テーブル エントリの行ステータスを定義します。

pgmonPIDList

現在モニタされているプロセスのプロセス ID (PID) のリストを定義します。

pgmonStatusList

現在モニタされている各プロセスのステータスを定義します。

pgmonSeverity

テーブル エントリに割り当てられた重大度を定義します。

SystemEDGE エージェントによって送信されるトラップの詳細については、「コンセプト」の章および付録「プライベート企業トラップ」を参照してください。

[プロセス グループ モニタ]テーブルの表示

CA Virtual Assurance を使用して、管理対象エージェントの [プロセス グループ モニタ] テーブルの内容を表示できます。

CA Virtual Assurance の [プロセス グループ モニタ] テーブルを表示する方法

1. CA Virtual Assurance ユーザ インターフェースにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペインで [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバ詳細のページが表示されます。
4. [設定] をクリックし、次に [プロセス グループ モニタ] をクリックします。
[プロセス グループ モニタ] テーブルに既存のすべてのプロセス グループ モニタが表示されます。

プロセス グループ モニタリングの設定

[プロセス グループ モニタ] テーブルでエントリを追加、削除、または変更することで、SystemEDGE エージェントがモニタするプロセスおよびプロセス属性を制御できます。

以下のいずれかの方法で [プロセス グループ モニタ] テーブルを設定できます。

CA Virtual Assurance からのファイルベース設定の動的使用

CA Virtual Assurance では、管理しているシステムにインストールされているエージェントを **CA Virtual Assurance** ユーザ インターフェースから管理できます。エージェントの再起動が必要ないユーザ フレンドリーで安全なメカニズムを備えたモニタを動的に定義して送信する機能が備わっているため、これは設定の推奨された方法です。個々のエージェントのモニタを設定するか、または複数の管理対象システムで適用するモニタ ポリシーを定義できます。個々の設定を変更するか、設定ポリシーを適用すると、**CA Virtual Assurance** によりエージェントシステムに新しい設定ファイルが送信されます。**CA Virtual Assurance** は、データ センター全体におけるエージェント設定の中心点の役割を果たします。また、他の方法によって加えられた変更を上書きしたり、拒否することができます。

CA Virtual Assurance を使用したエージェントの設定の詳細については、**CA Virtual Assurance** のドキュメントを参照してください。

eHealth AdvantEDGE View、CA Spectrum、または CA NSM などの管理システムからの SNMP Set コマンドの動的使用

さまざまな管理システムで、**SNMP Set** コマンドによる **SystemEDGE** へのモニタの追加がサポートされます。**eHealth** では、**AdvantEDGE View** インターフェースを介してこの機能を利用できます。また、**CA NSM** ではエージェント ビューを介して設定が可能で、**CA Spectrum** では **OneClick** を介して設定が可能です。**CA Virtual Assurance** を介した動的設定により利用可能な機能はどれも、これらの方法では利用できません。

SNMP Set の設定の詳細については、該当する管理システム用のドキュメントを参照してください。

ローカル sysedge.cf ファイルでのセルフ モニタリング テーブルのエントリの手動指定

マニュアル設定では、特定のディレクティブを使用してローカル **sysedge.cf** ファイルにエントリを追加する必要があります。 **sysedge.cf** ファイルからエージェントを直接設定した場合、変更を適用するには完全な再起動が必要です。 直接設定するにはエージェント データ ディレクトリにある **sysedge.cf** ファイルを使用します。

このセクションでは、ローカル **sysedge.cf** ファイルでセルフ モニタリングを直接設定する方法について説明しますが、あらゆる種類の設定に必要なプロパティについて説明します。

watch procgroup ディレクティブ --[プロセス グループ モニタ]テーブルへのエントリの追加

watch procgroup ディレクティブでは、**sysedge.cf** ファイルの [プロセス グループ モニタ] テーブルに直接プロセス グループ モニタ エントリを追加できます。 引数は、 [プロセス グループ モニタ] テーブルの列を表します。

以下で説明する構文を使用してエージェント データ ディレクトリ内の **sysedge.cf** ファイルに行を追加し、ファイルを保存し、エージェントを再起動して変更を適用します。

次のように、**watch procgroup** ディレクティブを使用して [プロセス グループ モニタ] テーブルにエントリを追加します。

```
watch procgroup 'regexpr' index flags interval ['descr'] ['action'] [severity]
'regexpr'
```

モニタするグループのプロセス ID を取得しようとするときに使用する正規表現を指定します。

index

このエントリに使用するモニタ テーブルの行 (インデックス) を指定します。 テーブル内の各行は、インデックス番号によって一意に識別されます。 行 1 ~ 10 はエージェントによる内部使用に予約済みです。したがって、インデックス値は 10 より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば **0x00000001**）を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグの詳細については、「プロセス (372P.)グループ モニタ」テーブルのフラグ」を参照してください。

interval

モニタを実行する必要がある頻度を秒単位で指定します。たとえば、値 **30** は、このエントリを **30** 秒ごとにモニタするようにエージェントに指示します。最小値は **30** で、**30** 秒の倍数で指定する必要があります。

'descr'

テーブルエントリの任意の説明 (**0** ～ **512** 文字の長さ)を指定します。

'action'

プロセス グループに一致が見つかったときに実行するコマンド (**0** ～ **2048** 文字の長さ)を、フルパスおよびパラメータを含めて指定します。文字列が空の場合、エージェントはこのエントリのアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定を変更できます。詳細については、「エージェントの設定」の章を参照してください。

severity

グループが変化してトラップが送信されたときにエントリへ割り当てる重大度を指定します。重大度はトラップと共に含まれています。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

重大度の指定は、モニタの重要性を指定するにすぎず、ステータスの計算には使用されない点に注意してください。

デフォルト: なし

watch procgroupex ディレクティブ --[プロセス グループ モニタ]テーブルへのエントリの追加 (UNIX のみ)

`watch procgroupex` ディレクティブでは、`sysedge.cf` ファイルの [プロセス グループ モニタ] テーブルに直接プロセス グループ モニタ エントリを追加できます。引数は、[プロセス グループ モニタ] テーブルの列を表します。

以下で説明する構文を使用してエージェント データ ディレクトリ内の `sysedge.cf` ファイルに行を追加し、ファイルを保存し、エージェントを再起動して変更を適用します。

次のように、`watch procgroupex` ディレクティブを使用して [プロセス グループ モニタ] テーブルにエントリを追加します。

```
watch procgroupex 'regexpr' user group index flags interval ['descr'] ['action']  
[severity]
```

regexpr

モニタするグループのプロセス ID を取得しようとするときに使用する正規表現を指定します。

user

プロセス名正規表現 (UNIX 固有) に加えて一致させるユーザ名を指定します。

group

プロセス名正規表現 (UNIX 固有) に加えて一致させるグループ名を指定します。

index

このエントリに使用するモニタ テーブルの行 (インデックス) を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 1 ~ 10 はエージェントによる内部使用に予約済みです。したがって、インデックス値は 10 より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値 (たとえば `0x00000001`) を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグの詳細については、「プロセス (372P.) グループ モニタ」テーブルのフラグ」を参照してください。

interval

モニタを実行する必要がある頻度を秒単位で指定します。たとえば、値 **30** は、このエントリを **30** 秒ごとにモニタするようにエージェントに指示します。最小値は **30** で、**30** 秒の倍数で指定する必要があります。

'descr'

テーブルエントリの任意の説明 (**0** ～ **512** 文字の長さ) を指定します。

'action'

プロセス グループに一致が見つかったときに実行するコマンド (**0** ～ **2048** 文字の長さ) を、フルパスおよびパラメータを含めて指定します。文字列が空の場合、エージェントはこのエントリのアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定を変更できます。詳細については、「エージェントの設定」の章を参照してください。

severity

グループが変化してトラップが送信されたときにエントリへ割り当てる重大度を指定します。重大度はトラップと共に含まれています。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

重大度の指定は、モニタの重要性を指定するにすぎず、ステータスの計算には使用されない点に注意してください。

デフォルト: なし

プロセス グループ モニタリングの例

このセクションには、プロセス グループ モニタリング用のサンプル設定ファイルディレクティブが記載されています。これらのエントリは、`sysedge.cf` ファイルで `watch procgroup` を使用して追加できます。

例: xterm プロセス グループのモニタ

以下の例では、`xterm` プロセス グループをモニタするようにエージェントを設定します。

```
watch procgroup 'xterm.*' 101 0x00 60 'Watch xterms' '' warning
```

101

このエントリが、[プロセス グループ モニタ] テーブルの行 **101** (`pgmonIndex=101`) に挿入されることを示します。

0x00

デフォルト動作を示します。

60

エージェントが **60** 秒ごとに `xterm` プロセス グループを確認することを示します。

warning

プロセス グループ変更トラップに警告重大度が含まれることを示します。

アクションは指定されないため、トラップを送信するとき、エージェントはコマンドを呼び出しません。

例:emacs プロセス グループのモニタ

以下の例では、emacs プロセス グループをモニタするようにエージェントを設定します。

```
watch procgroup 'emacs.*|xmibmgr' 12 0x00 60 'Watch emacs' '' major
```

12

このエントリが、[プロセス グループ モニタ] テーブルの行 12 (pgmonIndex=12) に挿入されることを示します。

0x00

デフォルト動作を示します。

60

エージェントが 60 秒ごとに emacs プロセス グループを確認するべきであることを示します。

major

プロセス グループ変更トラップにメジャー重大度が含まれることを示します。

アクションは指定されないため、トラップを送信するとき、エージェントはコマンドを呼び出しません。

例:DT プロセス グループのモニタ

以下の例では、DT プロセス グループをモニタするようにエージェントを設定します。

```
watch procgroup 'dt/bin' 13 0x00 60 'Watch DT stuff' ''
```

13

このエントリが、[プロセス グループ モニタ] テーブルの行 13 (pgmonIndex=13) に挿入されることを示します。

0x00

デフォルト動作を示します。

60

エージェントが 60 秒ごとに DT プロセス グループを確認することを示します。

アクションは指定されないため、トラップを送信するとき、エージェントはコマンドを呼び出しません。

関連項目

[Perl 互換正規表現 \(PCRE\) のサポート](#) (P. 178)

[テキスト パターン除外の設定](#) (P. 181)

[正規表現の例](#) (P. 181)

edgwatch ユーティリティ -- プロセス グループのモニタ

edgwatch コマンドライン ユーティリティは、SystemEDGE エージェントがプロセス、プロセス グループ、ログ ファイル、および Windows イベント ログをモニタするように自動的に設定します。特定のプロセス、プロセス グループ、ログ ファイル、または Windows イベント ログ、および関連付けられた引数を指定した後、edgwatch ユーティリティは SNMP Set リクエストを発行して、ターゲット エージェントの該当するモニタ テーブル内に該当するエントリを作成します。edgwatch ユーティリティは、エージェント インストールの bin サブディレクトリにあります。

プロセス グループ モニタリング要の edgwatch ユーティリティは、以下のように使用します。

```
edgwatch [-h hostname | ip_addr] [-p port] [-c community]
          [-v 1 | 2c | 3] [-u secName] [-s secLevel] [-n contextName]
          [-a authPassword] [-A MD5 | SHA]
          [-x privPassword] [-X DES | AES | 3DES]
          [-m FIPS_mode]
          [-r retries]
          [-t timeout] [-d logLevel] [-f logFile]
          -o procgrou command
```

-h *hostname* | *ipaddr*

エージェントが実行されているシステムのホスト名または IP アドレスを指定します。IPv4 アドレスおよび IPv6 アドレスを指定できます。

デフォルト: **localhost**

-p *port*

エージェントが実行されている UDP ポート (たとえば 1691) を指定します。

デフォルト: **161**

-c *community*

edgwatch がエージェントに対する **SNMP** リクエストで使用するコミュニティ文字列を指定します。**SNMPv1** および **SNMPv2c** で有効です。

デフォルト : **public**

-v {1 | 2c | 3}

エージェントが実行している **SNMP** のバージョン。**SNMPv1** の場合は **1** を、**SNMPv2c** の場合は **2c** を、**SNMPv3** の場合は **3** をそれぞれ指定します。

デフォルト : **1**

-u *secName*

SNMPv3 セキュリティに使用される **User-based Security Model (USM)** のユーザ名を指定します。

デフォルト : **none**

-s *secLevel*

以下に示した **SNMPv3** 通信用セキュリティ レベルのいずれかを指定します。

- 1 - noAuthNoPriv

- 2 - AuthNoPriv

- 3 - AuthPriv (**SNMPv3** のみ)

-n *contextName*

MIBMuxed エージェントのインスタンス名を指定します。

デフォルト : none

-a *authPassword*

エージェントが **secLevel 2 (AuthNoPriv)** または **3 (AuthPriv)** の **SNMPv3** に設定されている場合に、認証パスワードを指定します。

注: **SNMPv3** 通信では、このオプションの指定は不要です。

デフォルト : なし

-A {MD5 | SHA}

エージェントが **secLevel 2 (AuthNoPriv)** または **3 (AuthPriv)** の **SNMPv3** に設定されている場合に使用する認証プロトコルを指定します。現在、**MD5 (Message Digest Algorithm)** および **SHA (Secure Hash Algorithm)** のみを使用できます。

デフォルト: **MD5**

-x privPassword

エージェントが **secLevel 3 (AuthPriv)** の **SNMPv3** に設定されている場合に、プライバシー (暗号化) パスワードを指定します。

デフォルト: なし

-X {DES | AES | 3DES}

SNMPv3 ユーザが **secLevel 3 (AuthPriv)** で設定されている場合に、プライバシー プロトコルを指定します。データ暗号化標準の場合は **DES**、**128** ビットの暗号化キーを使用した高度暗号化標準 (**AES128**) の場合は **AES**、トリプルデータ暗号化標準の場合は **3DES** と指定します。

デフォルト: なし

-m FIPS_mode

操作の **FIPS** モードを制御します。許可される値は **0**、**1**、および **2** です。

0

非 **FIPS** モード

1

FIPS 共存モード

2

FIPS のみのモード

デフォルト: **1**

-r retries

再試行の回数を指定します。

デフォルト: **10**

-t timeout

SNMP レシーバが要求をタイムアウトしたと判断するまでの期間を指定します。

デフォルト : **10 秒**

-d logLevel

SNMP メッセージのログ レベルを指定します。許可される値は **0 ~ 5** です。

0

致命的なメッセージをログ記録します。

1

重大なメッセージをログ記録します。

2

警告メッセージをログ記録します。

3

情報メッセージをログ記録します。

4

すべてのメッセージをログ記録します。

5

デバッグ メッセージを含むすべてのメッセージをログ記録します。

デフォルト : **0**

-f logfile

エラーとデバッグ情報が含まれるログ ファイルの名前を指定します。

デフォルト : **sysedge_utility.log**

コマンド

コマンドおよび関連する引数を指定します。サポートされているコマンドには以下のものがあります。

- add
- setstatus
- delete
- list

注: これらのコマンドの詳細については、「プロセス グループ モニタリング用の edgewatch コマンド」を参照してください。

プロセス グループ モニタリング用の edgewatch コマンド

edgewatch のプロセス グループ モニタリング コマンドおよび関連付けられた引数を以下に示します。

```
add attribute regexpr index flags interval 'description' 'action'
```

```
setstatus index status
```

```
delete index
```

```
list
```

注: 太字の引数は、表示されたとおりに正確に入力する必要がある文字列リテラルです。他の引数は、ユーザが入力する必要がある値です。

regexpr

モニタするプロセスの **PID** を検索するために使用する正規表現を指定します。スペースまたは他の特殊文字が含まれている場合は、この値を引用符で囲みます。

index

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 **1 ～ 10** はエージェントによる内部使用に予約済みです。したがって、インデックス値は **10** より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば **0x00000001**）を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグのリストについては、「[プロセス グループ (372P.) モニタ] テーブルのフラグ」を参照してください。

interval

モニタを実行する必要がある頻度を秒単位で指定します。たとえば、値 **30** は、このエントリを **30** 秒ごとにモニタするようにエージェントに指示します。最小値は **30** で、**30** 秒の倍数で指定する必要があります。

'descr'

テーブル エントリの任意の説明 (**0 ～ 512** 文字の長さ) を指定します。

'action'

プロセス グループに一致が見つかったときに実行するコマンド (**0 ～ 2048** 文字の長さ) を、フルパスおよびパラメータを含めて指定します。文字列が空の場合、エージェントはこのエントリのアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定を変更できません。詳細については、「エージェントの設定」の章を参照してください。

status

エントリのステータスを定義します。以下のいずれかになります。

- active (行をアクティブにする)
- notInService (行を非アクティブにするが保持する)
- destroy (行を削除する)

edgewatch の例

以下のコマンドでは、インデックスが 10、11、12、および 13 の 4 つのモニタを定義します。インデックスが 10 および 11 のモニタにはオブジェクト情報が指定されていません。インデックスが 12 および 13 のモニタにはオブジェクト情報（クラス、インスタンス、属性）が指定されています。

```
edgewatch -c admin -o process add procAlive "lsold1" 10 0x00 60 "monitor oldls2" ""
edgewatch -c admin -o process add procSize "lsold2" 11 0x00 60 absolute "!=" 100
"monitor oldls1" ""
edgewatch -c admin -o process add procAlive "ls" 12 0x00 60 "monitor ls1" "" testClass
testInst testAttr none
edgewatch -c admin -o process add procSize "lsold2" 13 0x00 60 absolute "!=" 100
"monitor oldls1" "" testClass testInst testAttr none
edgewatch -c admin -o process list
```

プロセス グループ モニタリング エントリの削除

プロセス グループのモニタを停止するには、[プロセス グループ モニタ] テーブルおよび `sysedge.cf` ファイルから該当するエントリを削除する必要があります。CA Virtual Assurance ユーザ インターフェースからポイント設定を使用してエントリを削除すると、両方行うことができます。手動で削除するには、両方の場所から別個の操作でエントリを削除する必要があります。

CA Virtual Assurance のプロセス グループ モニタ エントリを削除する方法

1. CA Virtual Assurance ユーザ インターフェースにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペインで [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバ詳細のページが表示されます。
4. [設定] をクリックし、次に [プロセス グループ モニタ] をクリックします。
[プロセス グループ モニタ] テーブルに既存のすべてのセルフ モニタリングが表示されます。
5. 削除するモニタを選択し、[アクション]、[削除] を選択します。
エントリがテーブルから削除されます。エージェントが、ウォーム スタートを実行して変更を適用します。

プロセス グループ モニタ エントリを手動で削除する方法

1. ローカル `sysedge.cf` ファイルからエントリの `watch procgroup` ディレクティブを削除し、ファイルを保存します。
2. 以下のいずれかのコマンドで `-o` パラメータを使用して、`edgewatch` ユーティリティを実行します。

```
-o delete index
-o setstatus index 6
index
```

削除するプロセス グループ モニタ エントリのインデックス値を指定します。

`setstatus` コマンドを使用するとき、値を 6 にすると行ステータスが `destroy` に設定され、エントリが削除されます。

3. エージェントを再起動します。
エントリが削除されます。

Solaris ゾーン プロセス グループ モニタリング

フラグ `0x8000` (Solaris ゾーン プロセス) および変数 `pgmonRegExpr` の設定に応じて、`SystemEDGE` は Solaris ゾーン内のプロセス グループをモニタします。変数 `pgmonRegExpr` は、`ZoneRegExpr/ProcRegExpr` の形式で正規表現を指定します。互換性のため、変数 `pgmonZoneRegExpr` は引き続き利用できますが、`0x8000` フラグを使用して `sysedge.cf` からモニタが初期化されなかった場合のみ変更できます。

変数 `pgmonProcNameList` はすべてのプラットフォームで利用可能です。Solaris ゾーンがモニタされる場合、この変数には

`MatchedZoneName/MatchedProcessName` の形式のスペースで区切られた値のリストが含まれます。Solaris ゾーンのモニタが必要でないか、モニタできない場合 (Solaris 10 以外の OS)、この変数にはスペースで区切られた一致したプロセス名のリストが含まれます。

Solaris ゾーン内のプロセス グループをモニタするときは、フラグ `0x8000` 付きの `watch process` ディレクティブと `ZoneRegExpr/ProcRegExpr` の形式の正規表現を使用します。

アップグレード手順では、すべての `watch zone procgroup` ディレクティブが新しい形式に変換されます。

`SystemEDGE` は、すべてのプロセス モニタ関連トラップの `pgmonProcNameList` の値を送信します。

第 12 章: ログ ファイル モニタリング

この章では、SystemEDGE エージェントを使用して、正規表現についてログ ファイルを、サイズと内容についてディレクトリをモニタする方法について説明します。

このセクションには、以下のトピックが含まれています。

[ログ ファイル モニタリングの概要](#) (P. 393)

[ディレクトリ モニタリング](#) (P. 394)

[ログ モニタ テーブル](#) (P. 394)

[ログ モニタリングの設定](#) (P. 406)

[edgewatch ユーティリティ -- ログ ファイルのモニタ](#) (P. 411)

[ログ モニタリング エントリの削除](#) (P. 419)

[ログ ファイル モニタリングの推奨事項](#) (P. 420)

ログ ファイル モニタリングの概要

SystemEDGE エージェントでは、ユーザ指定の正規表現の表示方法について、UTF-8 エンコーディングテキスト ファイルを継続的にモニタできます。ログ ファイル モニタリングは、アプリケーションがログ記録するメッセージをモニタすることによってアプリケーションをモニタするための柔軟な解決策を提供します。この機能は、セキュリティ管理にも役立ちます。たとえば、セキュリティ違反の可能性を通知するメッセージがないかシステム ログ ファイルをモニタするように エージェントを設定できます。モニタするファイル、一致する正規表現、間隔、アクション、重大度、および他の値を指定するには、ログ モニタ テーブルを使用します。エージェントは定義されたログ ファイルを自動的にモニタし、正規表現の一致を検出すると、管理システムにトラップを送信します。

ログ ファイルの指定はワイルドカード表現である場合があります、そのため、エージェントはこの表現と一致する最近更新された単一のログ ファイルをモニタします。

エージェントが開始するとき(または行がログ モニタ テーブルに追加された後)、エージェントは、現在の長さとして最終アクセス日時について最近更新されたログ ファイルを識別して、ログ ファイル表現を評価します。その後、エージェントは、最後のステータス確認以降の追加または変更についてログ ファイル表現と一致する各ログ ファイルの統計を定期的に取ります。この動作により、指定されたエントリを手動で変更する必要がなく、1 つのモニタ エントリが名前を(場合によっては日付またはリビジョン情報と共に) 変更するログ ファイルに従うことができますようになります。

モニタされるログ ファイルが変更された場合、エージェントは、指定された正規表現に対する一致があるかどうかを確認するために、ログ ファイル全体ではなく、変更のみをスキャンします。エージェントが一致を検出した場合は、アクション フィールドが **NULL** でない限り、企業に固有の **logMonMatch SNMP** トラップを送信し、行に対して指定されたアクションを実行するようにエージェントを設定できます。

ディレクトリ モニタリング

ログ モニタ テーブルを使用して、内容とサイズについてファイル ディレクトリをモニタすることができます。ディレクトリのパスと名前を指定すると、エージェントがディレクトリのサイズとディレクトリ内のファイル数を追跡します。その後、ディレクトリのサイズと内容についてしきい値を設定するために、これらの属性についてセルフ モニタを定義することができます。

詳細については、「[ログ モニタ テーブルの列](#) (P. 395)」および「[watch logfile ディレクティブ](#) (P. 407)」を参照してください。

ログ モニタ テーブル

ログ モニタ テーブルは、SystemEDGE エージェントが現在モニタしているログ ファイルまたはディレクトリのそれぞれについての情報を提供します。行はそれぞれ、定義されたログ モニタ エントリを表します。

エントリごとに、ログ モニタ テーブルは以下の種類の情報を提供します。

- 特定の正規表現がないかエージェントがモニタしているログ ファイルの名前
- ログ ファイルの一致のための正規表現
- 一致が見つかったためにトラップが送信された回数
- 最後のトラップが送信された時間
- 直前の一致をもたらしたログ エントリ
- 重大度

ログ モニタ テーブルの列

以下のリストでは、ログ モニタ テーブルの列について説明します。ログ モニタ テーブルの詳細については、エージェント インストールの `mib` サブディレクトリ内の `Systems Management Empire MIB empire.asn1` を参照してください。

logMonitorIndex

このエントリの一意の行インデックスを定義します (**MAXINT** に対しては **1**)。行 **1** ~ **10** はエージェントの内部使用のために予約されています。追加の行のインデックスは **11** ~ **MAXINT** の範囲にする必要があります。

権限：読み取り専用

logMonitorLogFile

モニタされるログ ファイルまたはディレクトリの完全パスおよびファイル名を定義します。この値はワイルドカード表現にすることができ、最近更新されたログ ファイルを識別するためにスキャンごとにエージェントによって評価されます。

権限：読み書き

logMonitorRegularExpression

一致がないかログ ファイルをスキャンするときに検索する正規表現を定義します。**SystemEDGE** は、**512** 文字までの正規表現をサポートします。

権限：読み書き

logMonitorNumberTraps

正規表現と一致する文字列がファイルにログ記録されたため、エージェントがトラップ (**logMonMatchTrap**) を送信した回数を定義します。

権限：読み取り専用

logMonitorLastTrap

エージェントがこのエントリに最後にトラップ (**logMonMatchTrap**) を送信した時間を (**sysUpTime** に基づいて) 定義します。

権限：読み取り専用

logMonitorLastMatch

正規表現と一致した最後のログ ファイル エントリを定義します。この変数は一致が発生するたびに更新されます。

権限：読み書き

logMonitorStatus

行ステータスを定義します。以下のいずれかになります。

- active
- notInService
- notReady
- createAndGo
- createAndWait

通常、行は **active** または **notInService** のいずれかです。これらの値は、**SNMPv2** の **SMI RowStatus** テキスト表記法によって定義された値と意味が同一です。

権限：読み書き

デフォルト：**createAndWait(5)**

logMonitorLogFileSize

モニタされているファイルの現在のサイズ（バイト）またはモニタされているディレクトリの現在のサイズ（キロバイト）を定義します。

権限：読み取り専用

logMonitorLogFileLastUpdate

ファイルまたはディレクトリが最後に更新された時間を定義します。

権限：読み取り専用

logMonitorDescr

モニタされているファイルまたはディレクトリの説明（長さが **0** ～ **512** 文字）を定義します。

権限：読み書き

logMonitorAction

正規表現と一致しエージェントがトラップを送信したときに実行される引用符で囲まれたコマンド（長さが **0 ～ 2048** 文字）を、任意のパラメータと共に定義します。文字列が空の場合、エージェントはこのエントリのアクションを呼び出しません。

注: アクションに Windows バッチ ファイルを使用しないでください。プログラム上の厳しい制限があり、デスクトップアプリケーションでは正しく動作しないことがよくあります。代わりに、Perl または Visual Basic など、強力な柔軟なスクリプト言語を使用してください。

権限：読み書き

logMonitorFlags

エントリがその操作の過程で従う追加の動作セマンティクスを示す正の整数フラグを定義します。利用可能なフラグおよびフラグ設定の詳細については、「[ログ モニタ テーブルのフラグ \(P. 400\)](#)」を参照してください。

権限：読み書き

logMonitorMatches

正規表現と一致するログ ファイル エントリ の数を定義します。この値はトラップがこのエントリに送信されるかどうかにかかわらず増分します。この変数をポーリングすると、ユーザは一定時間の一致の割合または数を特定できます。

権限：読み取り専用

logMonitorInterval

ログ ファイル または ディレクトリ の連続するスキャンの最適な間隔（分単位）を定義します。

権限：読み書き

logMonitorLogFileName

logMonitorLogFile 属性にワイルドカード文字が含まれている場合に、モニタされている実際のファイル名またはディレクトリ名を定義します。

権限：読み取り専用

logMonitorLogFileCount

モニタされるディレクトリ内のファイルとサブディレクトリ数を定義します。ログ ファイルをモニタするとき、この値はファイルの存在に応じて **1** または **0** のいずれかです。

権限：読み取り専用

logMonitorSeverity

トラップが送信されたときにエントリに割り当てる重大度を定義します。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

デフォルト： **none**

行作成オブジェクト

ログ モニタ テーブルの **logMonIndex** 列は、テーブル内の行を識別するためのキーフィールド(または行インデックス)として機能します。行 1 ~ 10 は、SystemEDGE エージェントによる内部使用に予約済みです。行は 11 ~ MAXINT の範囲で設定できます。ログ モニタ テーブルを備えた以下の MIB オブジェクトを使用して、行作成を最適化することができます。

logmonUnusedIndex

オブジェクト上で **SNMP Get** を実行したときに、ログ モニタ テーブルの未使用のインデックス番号を返します。行作成に使用できる番号を取得するには、このオブジェクトをクエリします。

logmonMatchDescr

logmonDescr 属性内の特定のエントリの説明に対応するインデックス番号を特定します。**logmonDescr** 属性値を指定してこの MIB オブジェクト上で **SNMP Set** を実行すると、エージェントがログ モニタ テーブル内のエントリを検索し、**monMatchIndex** MIB オブジェクト内で最後に一致したエントリのインデックス値を配置します。

logmonMatchIndex

logmonMatchDescr と共に使用したときに、特定のエントリの説明をそのインデックス番号と照合します。

ローカル ポリシーに基づいて、行のブロックをシステム管理者に予約するように選択することができます。その後、その行が別のユーザのエントリによってすでに取得されている場合があることを懸念せずに、行の予約済みブロック内のエントリを定義できます。その他のユーザがユーザ設定のエントリを定義する場合は、ローカル ポリシーに従って、予約された範囲外にある行インデックスを使用する必要があります。

行のブロックを予約することによって、一貫した条件のセット(行エントリ)がすべてのマシンにおいてモニタされるよう定義することが可能となり、各マシンの同じ行番号に同じ条件が定義されるようになります。

CA Virtual Assurance でエージェントを設定すると、このプロセスを自動化できます。特定の行番号のエントリのブロックを設定し、1 回の操作ですべての管理対象システムにその設定を展開することができます。また、CA Virtual Assurance は、すでに使用されていて操作を拒否する行をユーザが定義しているかどうかを検出します。

CA Virtual Assurance 内での行の定義の詳細については、CA Virtual Assurance のドキュメントを参照してください。

ログ モニタ テーブルのフラグ

ログ モニタ テーブルの **logMonitorFlags** 列は、対応するログ モニタ テーブル行に追加の動作を指定できる、32 ビット符号なし整数です。デフォルトでは、ログ モニタ テーブル行は以下を実行します。

- 自分を再初期化しようとする
- SNMP トラップを送信する
- 一致したものを共通ログ ファイルにログ記録する
- アクションを呼び出す（アクションが設定されている場合）

これらのデフォルトを変更するように、さまざまなフラグ ビットを指定できます。**SystemEDGE** エージェントは、16 進（基数 16）法で表記されたすべてのフラグを解釈します。

ログ モニタ テーブルのフラグ（**logMonitorFlags**）を以下に示します。

フラグ値は 3 つのフィールドから構成されます。

- フィールド 1 には、**Systems Management Empire MIB** のモニタ テーブル用に定義された共通のテーブル フラグが含まれます。この部分はフラグの低位 8 ビットです。
- フィールド 2 には、各モニタ テーブル用に個別に定義された、テーブル固有のフラグが含まれます。このフィールドは、共通テーブル フラグの次の低位 12 ビットを定義します。

ログ モニタ テーブル用に 12 ビットがどのように定義されるかの詳細については、「[ログ モニタ テーブル \(P. 394\)](#)」の図を参照してください。

- フィールド 3 には、テーブル固有のフラグで使用する整数値用の予約済み高位 12 ビットが含まれます。このフィールドにはログ モニタ テーブルに固有のフラグが含まれます。

以下のセクションでは、各フラグ ビットを定義します。OR 論理演算によってフラグの値を組み合わせることができます。

以下のリストではログ モニタ テーブルのフラグについて説明します。

0x00000001

このエントリに対して、アクションが実行されないようにします。

0x00000002

このエントリに対して、**SNMP** トラップが送信されないようにします。

0x00000004

このエントリの再初期化を無効にします。デフォルトでは、モニタ対象のログ ファイルが常に利用不可能な場合、エージェントは定期的にこのテーブル エントリを再初期化しようとします。

0x00000008

sysedge.log ファイル内で、このエントリに対し、トラップのログが記録されないようにします。このビットを設定しても、トラップの送信には影響しません。イベントが頻繁に発生する場合、または特定のエントリをエージェントのハートビートとして使用している場合は、イベントのログ記録を無効にすると有用です。

0x00000010

エージェントがログ ファイル モニタリングを再初期化することを試みて失敗するたびに、このエントリの継続的な **logMonEntryNotReady** トラップを送信します。

エージェントのデフォルトの動作では、モニタ対象のログ ファイルが存在しなくなったとき、またはそのログ ファイルへのアクセス時にエラーが発生したときに、単一の **logMonEntryNotReady** トラップが送信されます。

エージェントは、定期的にエントリを再初期化しようとします。この機能を有効にすると、エージェントは、再初期化が失敗するたびに、**logMonEntryNotReady** トラップを送信します。

0x00000020

アクションスクリプトまたはプログラムへのデフォルト引数の受け渡しを無効にします。**SystemEDGE** は、通常、トラップタイプ、説明フィールドなどを示すデフォルト アクション パラメータを渡します。アクション パラメータの詳細については、「[ログ モニタ テーブルのアクション パラメータ \(P. 404\)](#)」を参照してください。

0x00000040

このエントリに対して **notReady** トラップが送信されないようにします。**notReady** トラップのログは記録されず、**notReady** トラップに対してアクションも実行されません。

0x00000100

正規表現の評価に **NOT** 論理演算子を適用します。このフラグを有効にすると、一致した正規表現は **false** と評価され、結果として、トラップは送信されず、アクションも実行されません。正規表現に一致しないすべてのエントリは **true** と評価され、結果としてトラップが送信され、指定された任意のアクションが実行されます。

0x00000200

ログ ファイルのサイズを追跡しますが、ファイル全体の解析は行いません。このフラグは、セルフ モニタリングと共に使用することにより、ログ ファイルの存在およびファイル サイズを追跡するのに役立ちます。

0xXXX00400

X 件の一致が発生するまで、モニタがイベントをトリガしないことを指定します。この値はスキャン単位で計算されます。このフラグを設定すると、一致の件数が **0x###00000** に格納された **X** の値を超えるまで、トラップ、イベント ログ記録、およびアクションが抑制されます。値を超えると、単一のイベントがトリガされます。このフラグは、イベントが頻繁に発生する場合に、トラップ トラフィックを最小化するのに役立ちます。

0x00000800

特定のログ ファイルの代わりにディレクトリをモニタするように指定します。

0x00001000

ディレクトリを再帰的にモニタします。

0x00002000

ディレクトリ モニタリング中にシンボリック リンクをたどらないように指定します。このフラグは、必ず **0x00000800** フラグ（ディレクトリ モニタリングのみ）と共に使用してください。

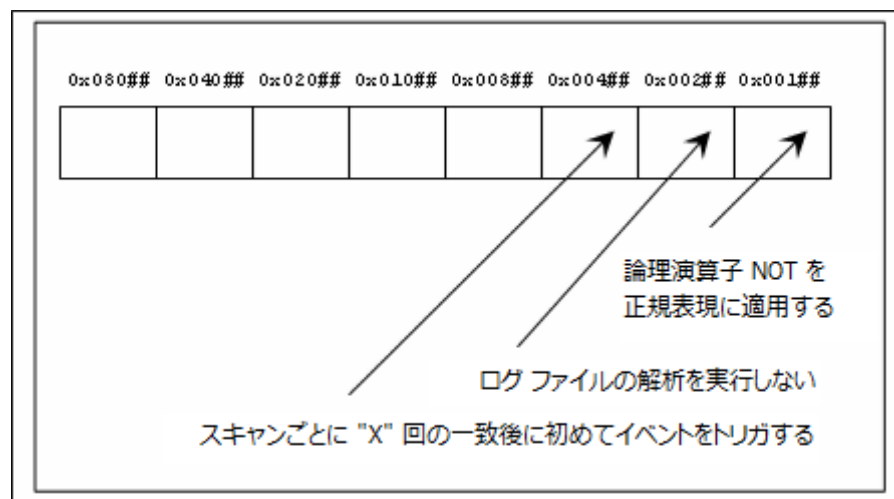
0x00004000

logMonitorLogFileSize 変数を、バイト単位ではなく、キロバイト単位でレポートします。

0x###00000

この **X** の値は **0x00000400** と共に使用され、イベントをトリガする前に必要なイベントの数を指定します。

以下の図は、ログ モニタ テーブルに固有のフラグ ビットを示します。



ログ モニタ テーブルのアクション パラメータ

SystemEDGE エージェントには、呼び出されたアクション コマンドに対していくつかのデフォルト パラメータが用意されています。これらのパラメータは、アクション文字列で指定したパラメータに加えられ、指定したパラメータの後のコマンドラインで渡されます。デフォルト パラメータは、ログ モニタ テーブルに対して送信される **SNMP** トラップ内に用意されているパラメータと同じです。

注: テーブル エントリで **0x00000020** フラグを設定することによって、エージェントがこれらのアクション パラメータをコマンドに含めないようにすることができます。

以下のリストでは、ログ モニタ テーブルのアクションのデフォルト パラメータについて説明します。

trapType

logMonMatchEvent や **logMonNotReadyEvent** など、送信されるトラップのタイプを定義します。

logMonitorLogFile

エージェントがモニタしているファイルの名前を定義します。

logMonitorRegularExpression

エージェントがエントリに対して照合しようとしている正規表現を定義します。

logMonitorLastTrap

このトラップが送信された時間を定義します。

logMonitorLastMatch

このトラップをトリガしたログ ファイルの行を定義します。

logMonitorDescr

テーブル エントリの説明を定義します。

logMonitorIndex

テーブル エントリのインデックスを定義します。

logMonitorFlags

テーブル エントリのフラグ フィールドを **16** 進法（たとえば **0x0000**）で定義します。

logMonitorLogFileName

logMonitorLogFile パラメータでワイルドカード文字が使用されている場合に、モニタされている実際のファイル名を定義します。

logMonitorSeverity

テーブル エントリに割り当てられた重大度を定義します。

ログ モニタ テーブルの表示

CA Virtual Assurance を使用して、管理対象エージェントのログ モニタ テーブルの内容を表示することができます。

CA Virtual Assurance でログ モニタ テーブルを表示する方法

1. CA Virtual Assurance ユーザ インターフェイスにアクセスします。
2. [リソース] タブをクリックします。
[エクスプローラ] ペインおよび[データ センター] ページが表示されます。
3. [エクスプローラ] ペイン内の[管理対象]を展開し、エージェントが存在するサーバを選択します。
サーバの詳細のページが表示されます。
4. [設定] をクリックし、[ログ モニタ] をクリックします。
ログ モニタ テーブルが、すべての既存のログ モニタと共に表示されます。

ログ モニタリングの設定

ログ モニタ テーブルでエントリを追加、削除、または変更することによって、SystemEDGE エージェントがモニタするログ ファイルを制御することができます。

ログ モニタ テーブルは以下の方法で設定できます。

CA Virtual Assurance からのファイルベース設定の動的使用

CA Virtual Assurance では、管理しているシステムにインストールされているエージェントを **CA Virtual Assurance** ユーザ インターフェースから管理できます。エージェントの再起動が必要ないユーザ フレンドリーで安全なメカニズムを備えたモニタを動的に定義して送信する機能が備わっているため、これは設定の推奨された方法です。個々のエージェントのモニタを設定するか、または複数の管理対象システムで適用するモニタ ポリシーを定義できます。個々の設定を変更するか、設定ポリシーを適用すると、**CA Virtual Assurance** によりエージェントシステムに新しい設定ファイルが送信されます。**CA Virtual Assurance** は、データ センター全体におけるエージェント設定の中心点の役割を果たします。また、他の方法によって加えられた変更を上書きしたり、拒否することができます。

CA Virtual Assurance を使用したエージェントの設定の詳細については、**CA Virtual Assurance** のドキュメントを参照してください。

eHealth AdvantEDGE View、CA Spectrum、または CA NSM などの管理システムからの SNMP Set コマンドの動的使用

さまざまな管理システムで、**SNMP Set** コマンドによる **SystemEDGE** へのモニタの追加がサポートされます。**eHealth** では、**AdvantEDGE View** インターフェースを介してこの機能を利用できます。また、**CA NSM** ではエージェント ビューを介して設定が可能で、**CA Spectrum** では **OneClick** を介して設定が可能です。**CA Virtual Assurance** を介した動的設定により利用可能な機能はどれも、これらの方法では利用できません。

SNMP Set の設定の詳細については、該当する管理システム用のドキュメントを参照してください。

ローカル sysedge.cf ファイルでのセルフ モニタリング テーブルのエントリの手動指定

マニュアル設定では、特定のディレクティブを使用してローカル **sysedge.cf** ファイルにエントリを追加する必要があります。 **sysedge.cf** ファイルからエージェントを直接設定した場合、変更を適用するには完全な再起動が必要です。直接設定するにはエージェント データ ディレクトリにある **sysedge.cf** ファイルを使用します。

このセクションでは、ローカル **sysedge.cf** ファイルでセルフ モニタリングを直接設定する方法について説明しますが、あらゆる種類の設定に必要なプロパティについて説明します。

watch logfile ディレクティブ -- ログ モニタ テーブルへのエントリの追加

watch logfile ディレクティブを使用すると、**sysedge.cf** ファイルでログ モニタ テーブルにログ モニタ エントリを直接追加できます。引数はログ モニタ テーブルの列を表します。

以下で説明する構文を使用してエージェント データ ディレクトリ内の **sysedge.cf** ファイルに行を追加し、ファイルを保存し、エージェントを再起動して変更を適用します。

watch logfile ディレクティブを使用して、ログ モニタ テーブルにエントリを以下のように追加します。

```
watch logfile index flags 'file' 'regexpr' ['descr'['action'[interval[severity]]]]
index
```

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 1 ~ 10 はエージェントによる内部使用に予約済みです。したがって、インデックス値は 10 より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば **0x00000001**）を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグの詳細については、「[ログ モニタ テーブルのフラグ \(P. 400\)](#)」を参照してください。

'file'

モニタするログ ファイルの完全パスとファイル名、またはモニタするディレクトリの完全パスを指定します。このパラメータは、*や?などのワイルドカード文字を含めて指定できます

モニタするファイルは **ASCII** ベースのテキスト ファイルである必要があります。**SystemEDGE** は、**Unicode** などのほかの文字セットのモニタリングをサポートしません。

テキスト エディタでファイルを開き [名前を付けて保存] を選択することにより、ファイルのエンコーディングを決定することができます。エンコーディングは [ファイルの種類] フィールドにリスト表示されます。

'regexpr'

一致がないかログ ファイルをスキャンするときに使用する正規表現を指定します。**SystemEDGE** は、**512** 文字までのログ ファイルの正規表現をサポートします。このパラメータは、ディレクトリをモニタしているときは無視されます。

'descr'

テーブル エントリの任意の説明 (**0** ～ **512** 文字の長さ) を指定します。

'action'

正規表現が一致しトラップが送信されたときに実行されるコマンド (長さが **0** ～ **2048** 文字) を、フルパスおよび任意のパラメータを含めて指定します。文字列が空の場合、エージェントはこのエントリのアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定を変更できます。詳細については、「エージェントの設定」の章を参照してください。

interval

このエントリをモニタする頻度を分単位で指定します。**interval** の有効な範囲は、**1** ～ **1440** 分です。

デフォルト: **1**

severity

一致が発生しトラップが送信されたときにエントリに割り当てる重大度を指定します。重大度はトラップと共に含まれています。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

重大度の指定は、モニタの重要性を指定するにすぎず、ステータスの計算には使用されない点に注意してください。

デフォルト: **none**

ログ ファイル モニタリングの例

このセクションでは、`sysedge.cf` ファイル内で `watch logfile` ディレクティブを使用して、ログ モニタ テーブルにログ モニタ エントリを追加する例を示します。

例: pop 接続の試行の検索

以下の例では、テーブル インデックス **15** でログ モニタ テーブルにエントリを追加して、システム上で **pop** 接続の試行を検索し、一致が発生したときに **warning** 重大度のトラップを送信するようにエージェントに指定します。

```
watch logfile 15 0x00 /var/log/syslog 'popper' 'NOTICE - pop connection' '' 1 warning
```

例: su の試行の検索

以下の例では、テーブル インデックス **16** でログ モニタ テーブルにエントリを追加して、システム上で **su** の試行を検索し、一致が発生したときに **critical** 重大度のトラップを送信するようにエージェントに指定します。

```
watch logfile 16 0x02 /var/adm/messages 'su.*fail' 'WARNING - su attempt'
'/local/bin/mail2admin' 5 critical
```

0x02

エージェントがトラップを送信しないように指定します。代わりに、エージェントは指定されたアクション コマンドを呼び出します。

例: ディレクトリのサイズと内容のモニタ

以下の例は、テーブルインデックス **17** でログ モニタ テーブルにエントリを追加して、ファイルのサイズと数について `/var/tmp` ディレクトリをモニタするようにエージェントに指定します。

```
watch logfile 17 0x00 /var/tmp 'Temporary directory' '' 5
```

セルフ モニタ エントリを作成して、サイズ (`logMonitorLogFileSize`) および内容 (`logMonitorLogFileCount`) 属性にしきい値を設定することができます。

以下の例は、テーブルインデックス **232** でログ モニタ テーブルにエントリを追加して、ファイルのサイズと数についてディレクトリをモニタするようにエージェントに指定します。

```
watch logfile 232 0x1800 'C:%testdir23' '' Monitor for dir testdir23' '' 1 warning
```

フラグ `0x0800` はディレクトリを非再帰的にモニタするために使用できます。これに加えて、セルフ モニタ エントリを作成して、サイズ (`logMonitorLogFileSize`) および内容 (`logMonitorLogFileCount`) 属性にしきい値を設定します。

```
monitor oid 1.3.6.1.4.1.546.11.1.1.8.232 2323232 0x8 30 absolute >= 10 'Directory Size  
is more than 10 bytes' '' 0 (or)  
monitor oid logMonitorLogFileSize.232 2323232 0x8 30 absolute >= 10 'Directory Size  
is more than 10 bytes' '' 0  
monitor oid logMonitorLogFileCount.232 2323231 0x8 30 absolute >= 10 'FileCount is  
more than 10' '' 0 (or)  
monitor oid 1.3.6.1.4.1.546.11.1.1.16.232 2323231 0x8 30 absolute >= 10 'File Count  
is more than 10' '' 0
```

例: 複数行でのパターンの検索

以下の例では、テーブルインデックス **20** でログ モニタ テーブルにエントリを追加するようにエージェントに指定する正規表現を指定します。そのエントリはモニタされるログ ファイル内の複数の行にわたって表示されるテキストを検索するために使用されます。この例では、テキスト「`Invalid login attempt`」の前に、テキスト「`WARNING:`」が未指定の行番号として表示されます。

```
watch logfile 20 0x00 /var/log/syslog '/^WARNING:(.*\n)*Invalid login attempt/m'  
'NOTICE - Invalid login attempt' '' 1 warning
```

注: この例では、PCRE (Perl Compatible Regular Expressions、Perl 互換正規表現) が、[ポリシー設定]-[制御設定]または `SystemEDGE` 設定ファイル(レガシー モードの場合)を使用して、エージェントでアクティブ化される必要があります。

関連項目

[Perl 互換正規表現 \(PCRE\) のサポート](#) (P. 178)

[テキスト パターン除外の設定](#) (P. 181)

[正規表現の例](#) (P. 181)

edgwatch ユーティリティ -- ログ ファイルのモニタ

edgwatch コマンドライン ユーティリティは、プロセス、プロセス グループ、ログ ファイル、および Windows イベント ログをモニタするように SystemEDGE エージェントを自動的に設定します。ユーザが特定のプロセス、プロセス グループ、ログ ファイル、または Windows イベント ログ、および関連する引数を指定すると、edgwatch ユーティリティはターゲット エージェントの適切なモニタリング テーブル内に適切なエントリを作成するための **SNMP Set** 要求を発行します。edgwatch ユーティリティはエージェント インストールの bin サブディレクトリにあります。

ログ ファイル モニタリング用の edgwatch ユーティリティの使用 방법은以下のとおりです。

```
edgwatch [-h hostname | ip_addr] [-p port] [-c community]
          [-v 1 | 2c | 3] [-u secName] [-s secLevel] [-n contextName]
          [-a authPassword] [-A MD5 | SHA]
          [-x privPassword] [-X DES | AES | 3DES]
          [-m FIPS mode]
          [-r retries]
          [-t timeout] [-d logLevel] [-f logFile]
          -o logfile command
```

-h *hostname* | *ipaddr*

エージェントが実行されているシステムのホスト名または IP アドレスを指定します。IPv4 アドレスおよび IPv6 アドレスを指定できます。

デフォルト: **localhost**

-p *port*

エージェントが実行されている UDP ポート (たとえば **1691**) を指定します。

デフォルト: **161**

-c community

エージェントが使用するコミュニティ文字列を指定します。 **SNMPv1** および **SNMPv2c** の場合のみ有効です。

注: snmpset に使用する読み取り/書き込みコミュニティ文字列を指定します。

デフォルト: **public**

-v {1 | 2c | 3}

エージェントが実行している **SNMP** のバージョン。 **SNMPv1** の場合は **1** を、**SNMPv2c** の場合は **2c** を、**SNMPv3** の場合は **3** をそれぞれ指定します。

デフォルト: **1**

-u secName

SNMPv3 セキュリティに使用される **User-based Security Model (USM)** のユーザ名を指定します。

デフォルト: **none**

-s secLevel

以下に示した **SNMPv3** 通信用セキュリティ レベルのいずれかを指定します。

- 1 - noAuthNoPriv
- 2 - AuthNoPriv
- 3 - AuthPriv (**SNMPv3** のみ)

-n contextName

MIBMuxed エージェントのインスタンス名を指定します。

デフォルト: **none**

-a authPassword

エージェントが **secLevel 2 (AuthNoPriv)** または **3 (AuthPriv)** の **SNMPv3** に設定されている場合に、認証パスワードを指定します。

注: **SNMPv3** 通信では、このオプションの指定は不要です。

デフォルト: なし

-A {MD5 | SHA}

エージェントが **secLevel 2 (AuthNoPriv)** または **3 (AuthPriv)** の **SNMPv3** に設定されている場合に使用する認証プロトコルを指定します。現在、**MD5 (Message Digest Algorithm)** および **SHA (Secure Hash Algorithm)** のみを使用できます。

デフォルト: **MD5**

-x *privPassword*

エージェントが **secLevel 3 (AuthPriv)** の **SNMPv3** に設定されている場合に、**プライバシー (暗号化) パスワード**を指定します。

デフォルト: なし

-X {DES | AES | 3DES}

SNMPv3 ユーザが **secLevel 3 (AuthPriv)** で設定されている場合に、**プライバシー プロトコル**を指定します。データ暗号化標準の場合は **DES**、**128 ビット**の暗号化キーを使用した高度暗号化標準 (**AES128**) の場合は **AES**、トリプルデータ暗号化標準の場合は **3DES** と指定します。

デフォルト: なし

-m *FIPS_mode*

操作の **FIPS** モードを制御します。許可される値は **0**、**1**、および **2** です。

0

非 **FIPS** モード

1

FIPS 共存モード

2

FIPS のみのモード

デフォルト: **1**

-r *retries*

再試行の回数を指定します。

デフォルト : **10**

-t *timeout*

SNMP レシーバが要求をタイムアウトしたと判断するまでの期間を指定します。

デフォルト : **10 秒**

-d *logLevel*

SNMP メッセージのログ レベルを指定します。許可される値は **0 ~ 5** です。

0

致命的なメッセージをログ記録します。

1

重大なメッセージをログ記録します。

2

警告メッセージをログ記録します。

3

情報メッセージをログ記録します。

4

すべてのメッセージをログ記録します。

5

デバッグ メッセージを含むすべてのメッセージをログ記録します。

デフォルト : **0**

-f *logfile*

エラーとデバッグ情報が含まれるログ ファイルの名前を指定します。

デフォルト : **sysedge_utility.log**

command

コマンドおよび関連する引数を指定します。サポートされているコマンドには以下のものがあります。

- add
- setstatus
- delete
- list

注: コマンドの詳細については、「[ログ ファイル モニタリング用の edgwatch コマンド](#) (P. 415)」を参照してください。

ログ ファイル モニタリング用の edgwatch コマンド

ログ ファイル モニタリング用の edgwatch コマンドおよび関連する引数は以下のとおりです。

```
add index flags "file" "regexpr" "descr" "action" interval
```

```
setstatus index status
```

```
delete index
```

```
list
```

index

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 1 ～ 10 はエージェントによる内部使用に予約済みです。したがって、インデックス値は 10 より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば 0x00000001）を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグの詳細については、「[ログ モニタ テーブルのフラグ](#) (P. 400)」を参照してください。

"file"

モニタするログ ファイルの完全パスとファイル名、またはモニタするディレクトリの完全パスを指定します。このパラメータは、*や?などのワイルドカード文字を含めて指定できます

モニタするファイルは **ASCII** ベースのテキスト ファイルである必要があります。**SystemEDGE** は、**Unicode** などのほかの文字セットのモニタリングをサポートしません。

テキスト エディタでファイルを開き [名前を付けて保存] を選択することにより、ファイルのエンコーディングを決定することができます。エンコーディングは [ファイルの種類] フィールドにリスト表示されます。

'regexpr'

一致がないかログ ファイルをスキャンするときに使用する正規表現を指定します。**SystemEDGE** は、**512** 文字までのログ ファイルの正規表現をサポートします。このパラメータは、ディレクトリをモニタしているときは無視されます。

"descr"

テーブル エントリの任意の説明 (長さが **0** ～ **512** 文字) を指定します。

"action"

正規表現が一致しトラップが送信されたときに実行されるコマンド (長さが **0** ～ **2048** 文字) を、フルパスおよび任意のパラメータを含めて指定します。文字列が空の場合、エージェントはこのエントリに対してアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定は変更できません。詳細については、「エージェントの設定」の章を参照してください。

interval

このエントリをモニタする頻度を分単位で指定します。

デフォルト : 1

status

ログ モニタ テーブルで行のステータスを設定する際に使用する、**RowStatus** テキスト表記法の値を指定します。このパラメータは **setStatus** 演算で使います。

有効な値は以下のとおりです。値は、割り当てられた整数値またはスペルアウトされた実際のステータス テキストのいずれかになります。

- active(1)
- notInService(2)
- destroy(6)

edgework の例

このセクションでは、SNMP バージョン 1、2c、および 3 で edgework ユーティリティを使用する例を示します。

例: ログ モニタ テーブル内のエントリのリスト

以下の例は、ログ モニタ テーブルの内容を表示します。

```
edgework -v 1 -h 127.0.0.1 -c public -o logfile list
```

```
edgework -v 2c -h fe80::2367:1 -c public -o logfile list
```

```
edgework -v 3 -s 3 -u userName -x privPassword -X encryptProtocol -A authProtocol  
-a authPassword -o logfile list
```

ログ モニタ エントリの追加

以下の例は、テーブルインデックス 5 でログ モニタ テーブルにエントリを追加して、HP-UX システム上に su エラーがないか検索するようにエージェントに指定します。エージェントは、一致を検索するとき、スクリプト `/local/bin/mail2admin` を実行します。

```
edgewatch -v 1 -h 127.0.0.1 -c private -o logfile add 5 0x00 /usr/adm/sulog "SU.* -"
"su attempt - WARNING" "/local/bin/mail2admin" 1
```

```
edgewatch -v 2c -h fe80::2367:1 -c private -o logfile add 5 0x00 /usr/adm/sulog "SU.*
-" "su attempt - WARNING" "/local/bin/mail2admin" 1
```

```
edgewatch -v 3 -h fe80::2367:1 -s 3 -u userName -A authProtocol -a authPassword -X
encryptProtocol -x privPassword -o logfile add 5 0x00 /usr/adm/sulog "SU.* -" "su
attempt - WARNING" "/local/bin/mail2admin" 1
```

例: ログ モニタ エントリの削除

以下の例では、テーブルインデックス 5 でログ モニタ テーブルからエントリを削除します。

```
edgewatch -v 1 -h 127.0.0.1 -c private -o logfile delete 5
```

```
edgewatch -v 2c -h fe80::2367:1 -c private -o logfile delete 5
```

```
edgewatch -v 3 -h 127.0.0.1 -s 3 -u userName -A authProtocol -a authPassword -X
encryptProtocol -x privPassword -o logfile delete 5
```

例: ログ モニタ エントリの無効化

以下の例では、エントリのステータスを `notInService(2)` に設定することにより、テーブルインデックス 5 でログ モニタ テーブルエントリを無効にします。エントリはテーブル内に残りますが、ステータスが `active(1)` に戻らない限り、エージェントは正規表現についてログ ファイルをスキャンしません。

```
edgewatch -v 1 -h 127.0.0.1 -c private -o logfile setstatus 5 notInService
```

```
edgewatch -v 2c -h fe80::2367:1 -c private -o logfile setstatus 5 notInService
```

```
edgewatch -v 3 -h 127.0.0.1 -s 3 -u userName -A authProtocol -a authPassword -X
encryptProtocol -x privPassword -o logfile setstatus 5 notInService
```

ログ モニタリング エントリの削除

ログ ファイルまたはディレクトリのモニタリングを停止するには、ログ モニタ テーブルおよび `sysedge.cf` ファイルから該当するエントリを削除する必要があります。CA Virtual Assurance ユーザ インターフェースからポイント設定を使用してエントリを削除すると、両方の場所からエントリを削除できます。手動の削除では、個別の操作で両方の場所からエントリを削除する必要があります。

CA Virtual Assurance 内でログ モニタ エントリを削除する方法

1. CA Virtual Assurance ユーザ インターフェースにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペイン内の [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバの詳細のページが表示されます。
4. [設定] をクリックし、[ログ モニタ] をクリックします。
ログ モニタ テーブルが、すべての既存のセルフ モニタと共に表示されます。
5. 削除するモニタを選択し、[アクション] - [削除] をクリックします。
テーブルからエントリが削除されます。エージェントは、変更を適用するためにウォーム スタートを実行します。

ログ モニタ エントリを手動で削除する方法

1. ローカルの `sysedge.cf` ファイルからエントリの `watch logfile` ディレクティブを削除し、ファイルを保存します。
2. `-o` パラメータの以下のいずれかのコマンドを使用して、`edgewatch` ユーティリティを実行します。

```
-o delete index  
-o setstatus index 6  
index
```

削除するログ モニタ エントリのインデックス値を指定します。

`setstatus` コマンドを使用するときに値を 6 に指定すると、行ステータスが破棄されるように設定され、エントリが削除されます。

3. エージェントを再起動します。
エントリが削除されます。

ログ ファイル モニタリングの推奨事項

システム ログとアプリケーション ログをモニタして、ユーザ、システム、およびアプリケーション動作に関する詳細な情報を取得することができます。

以下の表に、モニタできるログ ファイル、およびセキュリティ、デバイス エラー、システム容量、Windows セキュリティ、アプリケーションとシステムをモニタするために検索できる正規表現の推奨事項を記載します。

説明	モニタするログ ファイル	正規表現
WARNING - daemon core	/var/log/daemon-log	core dumped
WARNING - daemon core	/var/log/syslog	core dumped
Monitor SU attempts	/var/adm/messages	su.*fail
Monitor rlogins	/var/log/syslog	in.rlogin
Monitor telnets	/var/log/syslog	in.telnet
Monitor rsh	/var/log/syslog	in.rsh
WARNING - Illegal Instruction, Daemon	/var/log/daemon-log	.*illegal.*nstruction
Spam Relay Attempt	/var/log/syslog	Relaying denied
Monitor DENY packets from a Linux firewall	/var/log/messages	DENY

注: このセクションに記載したログ ファイルは、すべてのオペレーティング システムに対して同じではありません。これらの例は参考までに示したものです。オペレーティング システムに合わせて変更してください。

以下の表に、デバイス エラーをモニタするために、syslog (/var/adm/messages) 内で検索できる正規表現の推奨事項を記載します。

説明	正規表現
Critical: Badtrap Error	.*BAD TRAP.*
Error: SCSI error	.*SCSI.*[E,e]rror.*
Error: SCSI error	.*SCSI.*failed.*
Error: SCSI error	.*SCSI.*hung.*
Critical: badsimms error	.*SIMM.*

説明	正規表現
Critical: badsimms error	. *BAD. *SIMM.*
Critical: memory error	. *[M,m]emory [E,e]rror.*
Error: disk error	. *disk not responding.*
Error: disk error	. *[D,d]isk. *[E,e]rror.*
'Warning: disk fragmentation error'	. *optimization changed.*
Error: disk error	. *corrupt label.*
Error: I/O error	. *I/O. *[E,e]rror.*
Error: disk read/write errors	. *Error for Command:. *[read,write].*
Error: media error	. *Media Error.*
Info: serialport error	. *zs[0,1,2]: silo overflow.*
Warning: carrier error	. *no carrier.*
Warning: link is down	. *Link Down - cable problem?.*
Error: SDS error	. *[NOTICE,WARNING,PANIC]: md:.*

以下の表に、システム容量をモニタするために、/var/adm/messages 内で検索できる正規表現の推奨事項を記載します。

説明	正規表現
Critical: memory error	. *[O,o]ut of [M,m]emory.*
Critical: memory error	. *[F,f]ile system full.*
Error: disk space error	. *No space left on device.*

以下の表に、Windows セキュリティをモニタするために Windows イベント ログ内でモニタできるログと表現の推奨事項を記載します。

説明	イベント ログ	セキュリティ タイプ	正規表現
Random Password Hack	セキュリティ	すべて	. *bad
Misuse of Privileges	セキュリティ	すべて	. *[user rights,group management,security change, restart,shutdown]
Improper File Access	セキュリティ	障害	. *[read,write]

説明	イベント ログ	セキュリティ タイプ	正規表現
Improper Printer Access	セキュリティ	障害	.*print
Virus Outbreak Warning: program files updated	セキュリティ	すべて	.*write.*[exe,dll,com]
Security Policies Change	アプリケーション ン	情報	.*[S,s]ecurity policy

以下の表に、アプリケーションとシステムをモニタするために Windows イベント ログ内でモニタできるログと表現の推奨事項を記載します。

説明	イベント ログ	セキュリティ タイプ	正規表現
Application Error or Failure	アプリケーション ン	すべて	.*[F,f]ail .*[E,e]rror
Application Load Problems	アプリケーション ン	すべて	.*[L,l]oad.*[P,p]roblem
New Software Installed	アプリケーション ン	すべて	.*[INSTALL,Install,install]
Server Process failed during Initialization	アプリケーション ン	すべて	.*4131
Disk Failures and errors	すべて	すべて	.*[D,d]isk
Network Adapter Errors	すべて	エラー	.*[N,n]etwork [A,a]dapter

循環ログ ファイル

ログ ファイル エントリを指定するために、適切なワイルドカード エントリを使用して循環ログ ファイルをモニタすることができます。ワイルドカード 構文はプラットフォームに依存し、任意の数の文字に対してアスタリスク (*) を、任意の 1 文字を示すために疑問符 (?) を使用できます。エージェントは最新の一致するログ ファイルをモニタし、ログ ファイルが作成または更新されると、次の循環ログ ファイルに自動的に切り替えます。

第 13 章: Windows イベント モニタリング

この章では、SystemEDGE エージェントを使用して、正規表現について Windows イベント ログをモニタする方法について説明します。

重要: この章は、Windows バージョンの SystemEDGE エージェントにのみ関係します。

このセクションには、以下のトピックが含まれています。

[Windows イベント モニタリングの概要](#) (P. 423)

[NT イベント モニタ テーブル](#) (P. 425)

[Windows イベント モニタリングの設定](#) (P. 436)

[edgwatch ユーティリティ -- Windows イベントのモニタ](#) (P. 442)

[Windows イベント モニタリング エントリの削除](#) (P. 451)

Windows イベント モニタリングの概要

SystemEDGE エージェントでは、正規表現用について Windows イベント ログを継続的にモニタできます。この機能は、「ログ ファイル モニタリング」の章で説明されている標準的なログ ファイル モニタリングに似ています。ソースと説明、重大度、およびその他の値に対して照合するイベント ログ、イベント タイプ、正規表現を指定するには、NT イベント モニタ テーブルを使用します。エージェントは定義されたログを自動的にモニタし、正規表現の一致を検出すると、管理システムにトラップを送信します。

エージェントは、イベントをすぐに処理するアクション コマンドを実行することもできます。Windows イベントにはテキスト メッセージに加えていくつかの識別特性が含まれているため、このモニタリング機能は、指定可能な一致タイプでの標準的なログ ファイル モニタリングよりもやや複雑です。

SystemEDGE エージェントは、開始時に（または NT イベント モニタ テーブルへの行の追加後）、現在の長さとして最後に更新された時間について、各 Windows イベント ログのステータス（統計）を確認します。その後、エージェントは、最終更新以降の追加または変更について各イベント ログを定期的にスキャンします。イベント ログ ファイルが変更された場合、エージェントは、指定されたフィルタに対して一致が存在するかどうかを確認するために、イベント ログ全体ではなく、変更のみをスキャンします。

Windows イベントの検索条件

Windows イベント モニタ テーブル エントリはそれぞれ、以下の表に記載されている条件に基づいて一致がないか検索するようにエージェントに指定します。

イベント ログ

イベント ログの名前を指定します。この値は、以下のいずれかになります。

- アプリケーション
- システム
- セキュリティ
- DirService (ディレクトリ サービスの場合)
- DnsServer (DNS サービスの場合)
- FileRepService (ファイル レプリケーション サービスの場合)

イベント タイプ

イベントのタイプを指定します。タイプ 1 ～ 5 は **Windows** によって以下のように定義されます。

- error(1)
- warning(2)
- information(3)
- success(4)
- failure(5)

タイプ **all(6)** は、エージェントがすべてのイベント タイプと一致することを示します。

イベント ソース

イベントを生成したプログラムまたはモジュールの名前を指定します。エージェントは、このフィールドと一致する正規表現を使用します。

イベントの説明

イベントの説明です。エージェントは、このフィールドと一致する正規表現を使用します。

SystemEDGE エージェントは、4 つの条件すべてに基づいて一致を検索するときに、SNMP トラップ メッセージを生成します。この一致はブール AND ブール操作に似ています。

NT イベント モニタ テーブル

NT イベント モニタ テーブルは、エージェントが特定の正規表現パターンについて現在モニタしているイベント ログのそれぞれについての情報を提供します。行はそれぞれ、**Windows** イベント モニタ エントリを表します。

NT イベント モニタ テーブル内のエントリごとに、以下の種類の情報が提供されます。

- エージェントがモニタしているイベント ログ
- ログがモニタされている正規表現
- 一致が見つかったためにトラップが送信された回数
- 最後のトラップが送信された時間
- 直前の一致をもたらしたログ エントリ
- 重大度

NT イベント モニタ テーブルの列

以下のリストでは、NT イベント モニタ テーブルの列について説明します。NT イベント モニタ テーブルの詳細については、エージェント インストールの **mib** サブディレクトリ内の **Systems Management Empire MIB empire.asn1** を参照してください。

ntEventMonIndex

このエントリの一意の行インデックスを定義します (**MAXINT** に対しては **1**)。行 **1 ~ 10** はエージェントの内部使用に予約済みです。追加の行のインデックスは **11 ~ MAXINT** の範囲内である必要があります。

権限：読み取り専用

ntEventMonLog

モニタするイベント ログを定義します。有効な値は以下のとおりです。

- Application(1)
- Security(2)
- System(3)
- Directory Service(4)
- DNS Service(5)
- File Replication Service(6)

権限：読み書き

ntEventMonTime

イベントが発生した時間を **sysUpTime** に基づいて定義します。

権限：読み取り専用

ntEventMonTraps

このエントリに対する一致が発生し、エージェントがトラップ (**ntEventMonMatchTrap**) を送信した回数を定義します。

権限：読み取り専用

ntEventMonTypeLastMatch

検索条件と一致した最後のイベントのイベント タイプを定義します。タイプ 1 ～ 5 は **Windows** によって以下のように定義されます。

- error(1)
- warning(2)
- information(3)
- success(4)
- failure(5)

タイプ **noMatch(6)** は、このモニタリング エントリと一致するイベントがまだないことを示します。

権限：読み取り専用

ntEventMonTypeFilter

このエントリと一致するイベント タイプを定義します。タイプ 1 ～ 5 は **Windows** によって以下のように定義されます。

- error(1)
- warning(2)
- information(3)
- success(4)
- failure(5)

タイプ **all(6)** は、エージェントがすべてのイベント タイプと一致することを示します。

権限：読み書き

ntEventMonSrcLastMatch

このモニタ エントリと一致した最後のイベント ログ エントリのイベント ソースを定義します。 イベント ソースは通常、イベントを生成したプログラムの名前です。 一致が発生するたびに、この変数は更新されます。

権限： 読み取り専用

ntEventMonSrcFilter

一致がないかイベントをスキャンするときにイベント ソースに適用する正規表現を定義します。

権限： 読み書き

ntEventMonDescLastMatch

このモニタ エントリと一致した最後のイベント ログ エントリを定義します。 一致が発生するたびに、この変数は更新されます。

権限： 読み取り専用

ntEventMonDescFilter

一致がないかイベントをスキャンするときにイベントの説明に適用する正規表現を定義します。

権限： 読み書き

ntEventMonStatus

行ステータスを定義します。 行ステータスは以下のいずれかになります。

- active
- notInService
- notReady
- createAndGo
- createAndWait

通常、行は **active** または **notInService** のいずれかです。これらの値は、**SNMPv2** の **SMI RowStatus** テキスト表記法によって定義された値と意味が同じです。

権限： 読み書き

デフォルト： **createAndWait(5)**

ntEventMonDescr

モニタされているイベントの説明（長さが 0 ～ 512 文字）を定義します。

権限： 読み書き

ntEventMonAction

一致が検出されトラップが送信されたときに実行される引用符で囲まれたコマンド（長さが 0 ～ 2048 文字）を、任意のパラメータと共に定義します。文字列が空の場合、エージェントはこのエントリに対してアクションを呼び出しません。

注：アクションに対して Windows バッチ ファイルは使用しないでください。Windows バッチ ファイルは、プログラム上の厳しい制限を課し、デスクトップアプリケーションで正しく動作しない場合があります。代わりに、Perl や Visual Basic など、より強力な柔軟なスクリプト言語を使用してください。

権限： 読み書き

ntEventMonFlags

エントリが操作中に従う追加動作セマンティックを示す正の整数フラグを定義します。利用可能なフラグおよびフラグ設定の詳細については、「[NT イベント モニタ テーブルのフラグ \(P. 431\)](#)」を参照してください。

ntEventMonSeverity

トラップが送信されたときにエントリに割り当てる重大度を定義します。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

デフォルト： なし

行作成オブジェクト

NT イベント モニタ テーブルの **ntEventMonIndex** 列は、テーブル内の行を識別するためのキー フィールド (または行インデックス) として機能します。行 1 ~ 10 は、SystemEDGE エージェントによる内部使用に予約済みです。行は 11 ~ MAXINT の範囲で設定できます。

NT イベント モニタ テーブルを備えた以下の MIB オブジェクトを使用して、行作成を最適化することができます。

ntEventUnusedIndex

オブジェクト上で **SNMP Get** を実行したときに、NT イベント モニタ テーブルの未使用のインデックス番号を返します。行作成に使用できる番号を取得するには、このオブジェクトをクエリします。

ntEventMatchDescr

ntEventMonDescr 属性内の特定のエントリの説明に対応するインデックス番号を特定します。**ntEventMonDescr** 属性値を指定してこの MIB オブジェクト上で **SNMP Set** を実行すると、エージェントがセルフ モニタ テーブル内のエントリを検索し、**monMatchIndex** MIB オブジェクト内で最後に一致したエントリのインデックス値を配置します。

ntEventMatchIndex

ntEventMatchDescr と共に使用したときに、特定のエントリの説明をそのインデックス番号と照合します。

ローカル ポリシーに基づいて、行のブロックをシステム管理者に予約するように選択することができます。その後、その行が別のユーザのエントリによってすでに取得されている場合があることを懸念せずに、行の予約済みブロック内のエントリを定義できます。その他のユーザがユーザ設定のエントリを定義する場合は、ローカル ポリシーに従って、予約された範囲外にある行インデックスを使用する必要があります。

行のブロックを予約することによって、一貫した条件のセット（行エントリ）がすべてのマシンにおいてモニタされるよう定義することが可能となり、各マシンの同じ行番号に同じ条件が定義されるようになります。たとえば、行 11（monIndex = 11）を使用して swapCapacity 変数をモニタするためのエントリを定義することができます。また、この設定をすべてのシステムに配布できるため、行 11 はすべてのシステム上の swapCapacity 変数をモニタします。

CA Virtual Assurance でエージェントを設定すると、このプロセスを自動化できます。特定の行番号のエントリのブロックを設定し、1 回の操作ですべての管理対象システムにその設定を展開することができます。また、CA Virtual Assurance は、すでに使用されていて操作を拒否する行をユーザが定義しているかどうかを検出します。

CA Virtual Assurance 内での行の定義の詳細については、CA Virtual Assurance のドキュメントを参照してください。

NT イベント モニタ テーブルのフラグ

NT イベント モニタ テーブルの `ntEventMonFlags` 列を使用して、対応する NT イベント モニタ テーブル行に追加の動作セマンティックスを指定できます。デフォルトでは、NT イベント モニタ テーブル行は以下を実行します。

- 自分を再初期化しようとする
- SNMP トラップを送信する
- 共通ログ ファイルにイベントを記録する
- アクションを呼び出す（アクションが設定されている場合）

これらのデフォルトを変更するように、さまざまなフラグ ビットを設定できます。**SystemEDGE** エージェントは、16 進（基数 16）法で表記されたすべてのフラグを解釈します。

NT イベント モニタ テーブルのフラグ フィールド（`ntEventmonFlags`）を以下に示します。

フラグ値は 3 つのフィールドから構成されます。

- フィールド 1 には、**Systems Management Empire MIB** のモニタ テーブル用に定義された共通のテーブルフラグが含まれます。この部分はフラグの低位 8 ビットです。
- フィールド 2 には、各モニタ テーブル用に個別に定義された、テーブル固有のフラグが含まれます。このフィールドは、共通テーブルフラグの次の低位 12 ビットを定義します。**Windows** イベント モニタリングについては、テーブルに固有のフラグが現在定義されていません。
- フィールド 3 には、テーブル固有のフラグで使用する整数値用の予約済み高位 12 ビットが含まれます。**Windows** イベント モニタリングについては、テーブルに固有のフラグが現在定義されていません。

以下のセクションでは、各フラグ ビットを定義します。OR 論理演算によってフラグの値を組み合わせることができます。

以下のリストでは、モニタリング テーブル用の共通のテーブル フラグについて説明します。

0x00000001

このエントリに対して、アクションが実行されないようにします。

0x00000002

このエントリに対して、**SNMP** トラップが送信されないようにします。

0x00000004

このエントリの再初期化を無効にします。デフォルトでは、モニタ対象のイベント ログが常に利用できない場合、エージェントは定期的にこのテーブル エントリを再初期化しようとします。

0x00000008

sysedge.log ファイル内で、このエントリに対し、トラップのログが記録されないようにします。このビットを設定しても、トラップの送信には影響しません。イベントが頻繁に発生する場合、または特定のエントリをエージェントのハートビートとして使用している場合は、イベントのログ記録を無効にすると有用です。

0x00000010

エージェントがイベント ログ モニタリングを再初期化することを試みて失敗するたびに、このエントリの継続的な **ntEventMonEntryNotReady** トラップを送信します。エージェントのデフォルトの動作では、モニタ対象のイベント ログが存在しなくなったとき、またはそのイベント ログへのアクセス時にエラーが発生したときに、単一の **ntEventMonEntryNotReady** トラップが送信されます。エージェントは、定期的にエントリを再初期化しようとします。この機能を有効にすると、エージェントは、再初期化が失敗するたびに、**ntEventMonEntryNotReady** トラップを送信します。

0x00000020

アクション スクリプトまたはプログラムへのデフォルト引数の受け渡しを無効にします。**SystemEDGE** は、通常、トラップ タイプ、説明 フィールドなどを示すデフォルト アクション パラメータを渡します。アクション パラメータの詳細については、「[NT イベント モニタ テーブルのアクション パラメータ \(P. 434\)](#)」を参照してください。

0x00000040

このエントリに対して **notReady** トラップが送信されないようにします。**notReady** トラップのログは記録されず、**notReady** トラップに対してアクションも実行されません。

0x00000100

イベント ID 番号をイベント説明フィールドの先頭に追加します。識別子が [#] の形式を使用して追加されます。ここで、# は識別子です。このフラグの目的は、特定のイベント ID に関する検索を容易にすることです。たとえば、イベント ID 528 だけが一致するように指定するには、このフラグを指定し、ntEventMonDescFilter を [528] に設定します。

0x00000200

イベント ソース、タイプ、および説明の評価に、NOT 論理演算子を適用します。このフラグを有効にすると、指定したソース、タイプ、および説明に一致したイベントは **false** と評価され、結果として、トラップは送信されず、アクションも実行されません。イベント ソース、タイプ、および説明の 1 つ以上に一致しないイベントは **true** と評価され、結果としてトラップが送信され、指定された任意のアクションが実行されます。

注: このフラグを設定するときには、十分に注意してください。

0x####0000

NT イベント モニタ テーブルは、このフラグ値フィールドを使用しません。このフラグを設定しても、NT イベント モニタ テーブルの操作およびサポートされるいずれのフラグ ビットにも影響しません。

NT イベント モニタ テーブルのアクション パラメータ

SystemEDGE エージェントには、アクション コマンドに対していくつかのデフォルト パラメータが用意されています。これらのパラメータは、アクション文字列で指定したパラメータに加えられ、指定したパラメータの後のコマンドラインで渡されます。デフォルト パラメータは、NT イベント モニタ テーブルに対して送信される SNMP トラップ内に用意されているパラメータと同じです。

注: テーブル エントリで 0x00000020 フラグを設定することによって、エージェントがこれらのアクションパラメータをコマンドに含めないようにすることができます。

以下のリストでは、NT イベント モニタ テーブルのアクションのデフォルト パラメータについて説明します。

trapType

ntEventMonMatchEvent や **ntEventMonNotReadyEvent** など、送信されるトラップのタイプを定義します。

ntEventMonLog

エージェントがモニタしているイベント ログを定義します。

ntEventMonTypeLastMatch

このトラップを生成したイベントのタイプを定義します。

ntEventMonTime

このイベントが生成された時間を定義します。

ntEventMonSrcLastMatch

このトラップを生成したイベントのソースを定義します。

ntEventMonDescLastMatch

このトラップを生成したイベントの説明を定義します。

ntEventMonDesc

テーブル エントリの説明を定義します。

ntEventMonIndex

テーブル エントリ用のインデックスを定義します。

ntEventMonFlags

テーブルエントリのフラグ フィールドを 16 進法（たとえば 0x0000）で定義します。

ntEventMonSeverity

テーブルエントリに割り当てられた重大度を定義します。

SystemEDGE エージェントによって送信されるトラップの詳細については、「コンセプト」の章および付録「プライベート企業トラップ」を参照してください。

NT イベント モニタ テーブルの表示

CA Virtual Assurance を使用して、管理対象エージェントの NT イベント モニタ テーブルの内容を表示することができます。

CA Virtual Assurance で NT イベント モニタ テーブルを表示する方法

1. CA Virtual Assurance ユーザ インターフェイスにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペイン内の [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバの詳細のページが表示されます。
4. [設定] をクリックし、[Windows イベント モニタ] をクリックします。
Windows イベント モニタ テーブルが、すべての既存の Windows イベント モニタと共に表示されます。

Windows イベント モニタリングの設定

NT イベント モニタ テーブルでエントリを追加、削除、または変更することによって、SystemEDGE エージェントがモニタするイベント ログを制御することができます。

NT イベント モニタ テーブルは以下の方法で設定できます。

CA Virtual Assurance からのファイルベース設定の動的使用

CA Virtual Assurance では、管理しているシステムにインストールされているエージェントを **CA Virtual Assurance** ユーザ インターフェースから管理できます。エージェントの再起動が必要ないユーザ フレンドリーで安全なメカニズムを備えたモニタを動的に定義して送信する機能が備わっているため、これは設定の推奨された方法です。個々のエージェントのモニタを設定するか、または複数の管理対象システムで適用するモニタ ポリシーを定義できます。個々の設定を変更するか、設定ポリシーを適用すると、**CA Virtual Assurance** によりエージェントシステムに新しい設定ファイルが送信されます。**CA Virtual Assurance** は、データ センター全体におけるエージェント設定の中心点の役割を果たします。また、他の方法によって加えられた変更を上書きしたり、拒否することができます。

CA Virtual Assurance を使用したエージェントの設定の詳細については、**CA Virtual Assurance** のドキュメントを参照してください。

eHealth AdvantEDGE View、CA Spectrum、または CA NSM などの管理システムからの SNMP Set コマンドの動的使用

さまざまな管理システムで、**SNMP Set** コマンドによる **SystemEDGE** へのモニタの追加がサポートされます。**eHealth** では、**AdvantEDGE View** インターフェースを介してこの機能を利用できます。また、**CA NSM** ではエージェント ビューを介して設定が可能で、**CA Spectrum** では **OneClick** を介して設定が可能です。**CA Virtual Assurance** を介した動的設定により利用可能な機能はどれも、これらの方法では利用できません。

SNMP Set の設定の詳細については、該当する管理システム用のドキュメントを参照してください。

ローカル sysedge.cf ファイルでのセルフ モニタリング テーブルのエントリの手動指定

マニュアル設定では、特定のディレクティブを使用してローカル **sysedge.cf** ファイルにエントリを追加する必要があります。 **sysedge.cf** ファイルからエージェントを直接設定した場合、変更を適用するには完全な再起動が必要です。 直接設定するにはエージェント データ ディレクトリにある **sysedge.cf** ファイルを使用します。

このセクションでは、ローカル **sysedge.cf** ファイルでセルフ モニタリングを直接設定する方法について説明しますが、あらゆる種類の設定に必要なプロパティについて説明します。

watch ntevent ディレクティブ -- NT イベント モニタ テーブルへのエントリの追加

`watch ntevent` ディレクティブを使用すると、`sysedge.cf` ファイルで NT イベント モニタ テーブルに Windows イベント モニタ エントリを直接追加できます。引数は NT イベント モニタ テーブルの列を表します。

以下に記述する構文を使用してエージェント データ ディレクトリ内の `sysedge.cf` に行を追加し、ファイルを保存し、変更を有効にするためにエージェントを再起動します。

`watch ntevent` ディレクティブを使用して、NT イベント モニタ テーブルにエントリを以下のように追加します。

```
watch ntevent index flags 'evLog' 'evType' 'evSrc' 'evDescr' ['descr'] ['action']
[severity]
```

index

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 **1 ~ 10** はエージェントによる内部使用に予約済みです。したがって、インデックス値は **10** より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば `0x00000001`）を使用して、このエントリの動作に関する追加の指示を指定します。利用可能なフラグの詳細については、「[NT イベント モニタ テーブルのフラグ \(P. 431\)](#)」を参照してください。

'evLog'

モニタするイベント ログを指定します。値は、以下のいずれかになります。

- アプリケーション
- システム
- セキュリティ
- dnsServer
- dirService
- fileRepService

'evType'

このエントリと一致するイベント タイプを指定します。有効なタイプは、以下のとおりです。

- error
- warning
- information
- success
- failure
- all

「all」を入力すると、すべてのイベント タイプが一致することを示します。

'evSrc'

各イベントのイベント ソース属性をスキャンするときに使用する正規表現を指定します。

'evDescr'

各イベントのイベント説明属性をスキャンするときに使用する正規表現を指定します。

'descr'

テーブル エントリの 任意の説明 (0 ～ 512 文字の長さ) を指定します。

'action'

エントリが一致しトラップが送信されたときに実行されるコマンド (長さが 0 ～ 2048 文字) を、フルパスおよび任意のパラメータを含めて指定します。文字列が空の場合、エージェントはこのエントリに対してアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定は変更できません。詳細については、「エージェントの設定」の章を参照してください。

severity

一致が発生しトラップが送信されたときにエントリに割り当てる重大度を指定します。重大度はトラップに含まれます。有効な値は以下のとおりです。

- none
- ok
- warning
- minor
- major
- critical
- fatal

重大度の指定は、モニタの重要性を指定するだけで、ステータスの計算には使用されません。

デフォルト: なし

Windows イベント モニタリングの例

このセクションでは、`watch ntevent` ディレクティブを使用して Windows イベント ログをモニタするための、NT イベント モニタ テーブルのエントリ例を示します。これらのエントリは `sysedge.cf` ファイルに追加できます。

例: Web サーバ メッセージについてのアプリケーション ログの検索

以下の例では、`http` Web サーバ アプリケーションからのメッセージがないかアプリケーション ログを検索し、一致が発生したときに **warning** 重大度のトラップを送信するために、テーブル インデックス **11** でエージェントの NT イベント モニタ テーブルに新しいエントリを追加します。

```
watch ntevent 11 0x00 Application All 'http' '.*' 'Web Server messages' " warning
```

例: エラー イベントについてのセキュリティ ログの検索

以下の例では、ログイン エラーを示すエラー イベントがないかセキュリティ ログを検索し、一致が発生したときに **critical** 重大度のトラップを送信するために、テーブル インデックス **12** でエージェントの NT イベント モニタ テーブルに新しいエントリを追加します。

```
watch ntevent 12 0x00 Security Failure '.*' '.*' 'Access Failure - WARNING' " critical
```

例: 特定のイベントについてのアプリケーション ログの検索

以下の例では、イベント ID **277** のイベントがないかアプリケーション ログを検索するために、テーブル インデックス **3** でエージェントの NT イベント モニタ テーブルに新しいエントリを追加します。

```
watch ntevent 3 0x0100 Application All '.*' '¥[277¥]' 'Event ID 277' "
0x0100
```

説明にイベント ID を追加します。**[277¥]** は、エージェントが照合しようとする説明フィールドです。

角かっこ (`[]`) は正規表現の一致用の特殊文字であるため、円記号 (`¥`) が必要です。

関連項目

[Perl 互換正規表現 \(PCRE\) のサポート](#) (P. 178)

[テキスト パターン除外の設定](#) (P. 181)

[正規表現の例](#) (P. 181)

edgwatch ユーティリティ -- Windows イベントのモニタ

edgwatch コマンドライン ユーティリティは、プロセス、プロセス グループ、ログ ファイル、および Windows イベント ログをモニタするように SystemEDGE エージェントを自動的に設定します。ユーザが特定のプロセス、プロセス グループ、ログ ファイル、または Windows イベント ログ、および関連する引数を指定すると、edgwatch ユーティリティはターゲット エージェントの適切なモニタリング テーブル内に適切なエントリを作成するための **SNMP Set** 要求を発行します。edgwatch ユーティリティはエージェント インストールの bin サブディレクトリにあります。

Windows イベント モニタリング機能は Windows 上でのみ提供されますが、edgwatch ユーティリティを使用して、サポートされている任意のプラットフォームから設定できます。

Windows イベント モニタリング用の edgwatch ユーティリティの使用方法は以下のとおりです。

```
edgwatch [-h hostname | ip_addr] [-p port] [-c community]
          [-v 1 | 2c | 3] [-u secName] [-s secLevel] [-n contextName]
          [-a authPassword] [-A MD5 | SHA]
          [-x privPassword] [-X DES | AES | 3DES]
          [-m FIPS_mode]
          [-r retries]
          [-t timeout] [-d logLevel] [-f logFile]
          -o ntevent command
```

-h hostname | ipaddr

エージェントが実行されているシステムのホスト名または IP アドレスを指定します。IPv4 アドレスおよび IPv6 アドレスを指定できます。

デフォルト : **localhost**

-p port

エージェントが実行されている UDP ポート（たとえば **1691**）を指定します。

デフォルト : **161**

-c community

エージェントが使用するコミュニティ文字列を指定します。SNMPv1 および SNMPv2c の場合のみ有効です。

注: snmpset に使用する読み取り/書き込みコミュニティ文字列を指定します。

デフォルト : **public**

-v {1 | 2c | 3}

エージェントが実行している **SNMP** のバージョン。 **SNMPv1** の場合は **1** を、 **SNMPv2c** の場合は **2c** を、 **SNMPv3** の場合は **3** をそれぞれ指定します。

デフォルト： **1**

-u secName

SNMPv3 セキュリティに使用される **User-based Security Model (USM)** のユーザ名を指定します。

デフォルト： **none**

-s secLevel

以下に示した **SNMPv3** 通信用セキュリティ レベルのいずれかを指定します。

- 1 - noAuthNoPriv
- 2 - AuthNoPriv

3 - AuthPriv (SNMPv3 のみ)

-n contextName

MIBMuxed エージェントのインスタンス名を指定します。

デフォルト： **none**

-a authPassword

エージェントが **secLevel 2 (AuthNoPriv)** または **3 (AuthPriv)** の **SNMPv3** に設定されている場合に、認証パスワードを指定します。

注: **SNMPv3** 通信では、このオプションの指定は不要です。

デフォルト： なし

-A {MD5 | SHA}

エージェントが **secLevel 2 (AuthNoPriv)** または **3 (AuthPriv)** の **SNMPv3** に設定されている場合に使用する認証プロトコルを指定します。現在、**MD5 (Message Digest Algorithm)** および **SHA (Secure Hash Algorithm)** のみを使用できます。

デフォルト: **MD5**

-x privPassword

エージェントが **secLevel 3 (AuthPriv)** の **SNMPv3** に設定されている場合に、プライバシー (暗号化) パスワードを指定します。

デフォルト: なし

-X {DES | AES | 3DES}

SNMPv3 ユーザが **secLevel 3 (AuthPriv)** で設定されている場合に、プライバシー プロトコルを指定します。データ暗号化標準の場合は **DES**、**128** ビットの暗号化キーを使用した高度暗号化標準 (**AES128**) の場合は **AES**、トリプルデータ暗号化標準の場合は **3DES** と指定します。

デフォルト: なし

-m FIPS_mode

操作の **FIPS** モードを制御します。許可される値は **0**、**1**、および **2** です。

0

非 **FIPS** モード

1

FIPS 共存モード

2

FIPS のみのモード

デフォルト: **1**

-r retries

再試行の回数を指定します。

デフォルト: **10**

-t timeout

SNMP レシーバが要求をタイムアウトしたと判断するまでの期間を指定します。

デフォルト : **10 秒**

-d logLevel

SNMP メッセージのログ レベルを指定します。許可される値は **0 ~ 5** です。

0

致命的なメッセージをログ記録します。

1

重大なメッセージをログ記録します。

2

警告メッセージをログ記録します。

3

情報メッセージをログ記録します。

4

すべてのメッセージをログ記録します。

5

デバッグ メッセージを含むすべてのメッセージをログ記録します。

デフォルト : **0**

-f logfile

エラーとデバッグ情報が含まれるログ ファイルの名前を指定します。

デフォルト : **sysedge_utility.log**

command

コマンドおよび関連する引数を指定します。サポートされているコマンドには以下のものがあります。

- add
- setstatus
- delete
- list

注: コマンドの詳細については、「[Windows イベント モニタリング用の edgewatch コマンド](#) (P. 447)」を参照してください。

Windows イベント モニタリング用の edgework コマンド

edgework Windows イベント モニタリング コマンドおよび関連する引数は以下のとおりです。

```
add index flags evLog evType "evSrc" "evDesc" "descr" "action"
```

```
setstatus index status
```

```
delete index
```

```
list
```

index

このエントリに使用するモニタ テーブルの行（インデックス）を指定します。テーブル内の各行は、インデックス番号によって一意に識別されます。行 **1 ~ 10** はエージェントによる内部使用に予約済みです。したがって、インデックス値は **10** より大きく、テーブル全体で一意である必要があります。

flags

16 進法フラグ値（たとえば **0x00000001**）を使用して、このエントリの動作に関する追加の指示を指定します。

evLog

モニタするイベント ログを指定します。値は、以下のいずれかになります。

- application
- system
- security
- dnsServer
- dirService
- fileRepService

evType

このエントリと一致するイベント タイプを指定します。有効なタイプは、以下のとおりです。

- error
- warning
- information
- success
- failure
- all

「all」を入力すると、すべてのイベント タイプが一致することを示します。

"evSrc"

各イベントのイベント ソース属性をスキャンするときに使用する正規表現を指定します。

"evDescr"

各イベントのイベント説明属性をスキャンするときに使用する正規表現を指定します。

"descr"

テーブルエントリの任意の説明（長さが **0** ～ **512** 文字）を指定します。

"action"

エントリが一致しトラップが送信されたときに実行されるコマンド（長さが **0** ～ **2048** 文字）を、フルパスおよび任意のパラメータを含めて指定します。文字列が空の場合、エージェントはこのエントリに対してアクションを実行しません。

注: エージェントがアクションを実行するときのデフォルト設定は変更できます。詳細については、「エージェントの設定」の章を参照してください。

ステータス

setstatus 操作と共に使用されたときに NT イベント モニタ テーブルで行のステータスを設定する際に使用する、**Status** テキスト表記法の値を指定します。有効な値は以下のとおりです。値は、割り当てられた整数値またはスペルアウトされた実際のステータス テキストのいずれかになります。

- active(1)
- notInService(2)
- destroy(6)

edgewatch の例

このセクションでは、Windows イベントをモニタするために、SNMP バージョン 1、2c、および 3 で edgewatch ユーティリティを使用する例を示します。

例: NT イベント モニタ テーブル内のエントリのリスト

以下の例は、エージェントの NT イベント モニタ テーブルの内容をリストします。

```
edgewatch -v 1 -h fe80:ab01::901:bdef -c public -o ntevent list
```

```
edgewatch -v 2c -h 127.0.0.1 -c public -o ntevent list
```

```
edgewatch -v 3 -h fe80:ab01::901:bdef -s 3 -u userName -A authProtocol -a authPassword  
-X encryptProtocol -x privPassword -o ntevent list
```

例: NT イベント モニタ エントリの追加

以下の例では、Windows システム上のログイン エラーを検索するために、テーブル インデックス 5 でエージェントの NT イベント モニタ テーブルに新しいエントリを追加します。

```
edgewatch -v 1 -h 127.0.0.1 -c private -o ntevent add 5 0x0 Security Failure ".*" ".*" "Failed login attempt - WARNING" "%local%bin%mail2admin.exe"
```

```
edgewatch -v 2c -h fe80:ab01::901:bdef -c private -o ntevent add 5 0x0 Security Failure  
".*" ".*" "Failed login attempt - WARNING" "%local%bin%mail2admin.exe"
```

```
edgewatch -v 3 -h fe80:ab01::901:bdef -s 3 -u userName -A authProtocol -a authPassword  
-X encryptProtocol -x privPassword -o ntevent add 5 0x0 Security Failure ".*" ".*" "Failed login attempt - WARNING" "%local%bin%mail2admin.exe"
```

この例はまた、エージェントが一致を検出したときに %local%bin%mail2admin.exe スクリプトを実行するようにエージェントに指定します。

例: NT イベント モニタ エントリの削除

以下の例では、テーブル インデックス 5 でエージェントの NT イベント モニタ テーブルからエントリを削除します。

```
edgewatch -v 1 -h 127.0.0.1 -c private -o ntevent delete 5
```

```
edgewatch -v 2c -h fe80:ab01::901:bdef -c private -o ntevent delete 5
```

```
edgewatch -v 3 -h 127.0.0.1 -s 3 -u userName -A authProtocol -a authPassword -X  
encryptProtocol -x privPassword -o ntevent delete 5
```

例: NT イベント モニタ エントリの無効化

以下の例では、エントリのステータスを `notInService(2)` に設定することにより、テーブル インデックス 5 で NT イベント モニタ テーブル エントリを無効にします。エントリはテーブル内に残りますが、エントリのステータスが `active(1)` に戻らない限り、エージェントは一致についてイベント ログをスキャンしません。

```
edgewatch -v 1 -h 127.0.0.1 -c private ntevent setstatus 5 2
```

```
edgewatch -v 2c -h fe80:ab01::901:bdef -c private ntevent setstatus 5 2
```

```
edgewatch -v 3 -h 127.0.0.1 -s 3 -u userName -A authProtocol -a authPassword -X  
encryptProtocol -x privPassword -o ntevent setstatus 5 2
```

2

RowStatus テキスト表現法の値 `notInService(2)` に一致します。

Windows イベント モニタリング エントリの削除

Windows イベントのモニタリングを停止するには、NT イベント モニタ テーブルおよび `sysedge.cf` ファイルから該当するエントリを削除する必要があります。CA Virtual Assurance ユーザ インターフェースからポイント設定を使用してエントリを削除すると、両方の場所からエントリを削除できます。手動の削除では、個別の操作で両方の場所からエントリを削除する必要があります。

CA Virtual Assurance 内で Windows イベント モニタ エントリを削除する方法

1. CA Virtual Assurance ユーザ インターフェースにアクセスします。
2. [リソース] タブをクリックします。
[管理対象リソース] ペインおよび [データ センター] ページが表示されます。
3. [管理対象リソース] ペイン内の [管理対象] を展開し、エージェントが存在するサーバを選択します。
サーバの詳細のページが表示されます。
4. [設定] をクリックし、[NT イベント ログ モニタ] をクリックします。
NT イベント モニタ テーブルが、すべての既存のセルフ モニタと共に表示されます。
5. 削除するモニタを選択し、[アクション] - [削除] をクリックします。
テーブルからエントリが削除されます。エージェントは、変更を適用するためにウォーム スタートを実行します。

Windows イベント モニタ エントリを手動で削除する方法

1. ローカルの `sysedge.cf` ファイルからエントリの `nvent` ディレクティブを削除し、ファイルを保存します。
2. `-o` パラメータの以下のいずれかのコマンドを使用して、`edgewatch` ユーティリティを実行します。

```
-o delete index
```

```
-o setstatus index 6
```

```
index
```

削除する **Windows** イベント モニタ エントリのインデックス値を指定します。

`setstatus` コマンドを使用するときに値を **6** に指定すると、行ステータスが破棄されるように設定され、エントリが削除されます。

3. エージェントを再起動します。

エントリが削除されます。

第 14 章：履歴の収集

この章では、SystemEDGE エージェントの履歴サンプリング機能について説明し、後でマネージャが取得できるように、時間の経過と共に MIB 変数の値をモニタして格納するようにエージェントに指示する方法について説明します。

このセクションには、以下のトピックが含まれています。

[履歴の収集の概要](#) (P. 453)

[履歴制御テーブルと履歴データ テーブル](#) (P. 454)

[履歴サンプリングの例](#) (P. 458)

[履歴制御テーブルの表示](#) (P. 460)

[履歴制御テーブルの設定](#) (P. 460)

履歴の収集の概要

SystemEDGE は、後で取得できるように、時間の経過と共に整数ベースの各種 MIB オブジェクト（カウンタ、ゲージなど）の値を追跡して、格納できます。この機能は、一般に履歴サンプリングと呼ばれており、ネットワーク全体の管理ステーションのポーリング量を大幅に削減できます。

連続的にポーリングして時間の経過と共に MIB 変数の値を収集しなくても、マネージャは値をサンプリングして格納するようエージェントに指示することができます。管理システムは、エージェントに定期的に問い合わせ、サンプルの全履歴をアップロードすることができます。管理システムがエージェントと通信できないとき、ネットワークが停止していても、エージェントは指定された MIB 値をサンプリングして格納し続けることができます。

履歴サンプリング

エージェントは、次の 2 つの SNMP MIB テーブルを使用して履歴機能を提供します。

- データ収集機能を定義するための履歴制御テーブル。
- 実際のデータ サンプルを格納するための履歴テーブル。

制御テーブルでは、制御下の整数ベースの MIB 変数の値をサンプリングして格納するように動的にエージェントを設定できます。データ テーブルには、後で取得できるように値が格納されます。

ベースライン決定および傾向分析を実行するには、**swapCapacity** の値をモニタして格納するように **SystemEDGE** を設定します。たとえば、5 分おきに値をサンプリングして、最新の **144** のサンプルを格納するようにエージェントを設定することで行います。その場合、12 時間に 1 回、管理システムは、**144** のサンプルをすべてアップロードして、直近 12 時間に収集された **swapCapacity** の値を取得できます。

履歴制御テーブルと履歴データ テーブル

履歴制御テーブルには、エージェントがサンプリングして履歴テーブルに格納するデータを説明するパラメータが含まれています。履歴制御テーブルの各行では、テーブルのパラメータ（列のオブジェクト）に値が割り当てられているため、特定のデータ収集機能が定義されます。履歴テーブル内の 1 つ以上の行（格納されたサンプル）が、その 1 つの制御行と関連付けられます。

制御テーブルの各行には、**empireHistoryCtrlIndex** の一意の値が割り当てられます。行は、サンプリングするオブジェクトインスタンス、サンプリング間隔（30 秒の倍数）、および保持するサンプルの数（バケット）を指定することで、データ収集機能を定義します。各データ収集機能（制御テーブルの行）に関連付けられるのは、履歴テーブルの行のセットです。履歴テーブルの各行（バケットとも呼ばれます）には、1 つのサンプリング間隔に収集された指定された **MIB** オブジェクトの値が保持されます。

サンプリング間隔ごとに、エージェントは、このデータ収集機能の他の行と同じ **empireHistoryIndex** を使用して、履歴テーブルに新しい行を追加します。新しい各行は、履歴制御テーブル内の 1 つの行に対応しており、前のサンプルの **SampleIndex** より 1 つ大きい **empireHistorySampleIndex** が付与されます。

履歴制御テーブルの列

以下のリストでは、履歴制御テーブルの列について説明します。

empireHistoryCtrlIndex

テーブル内のエントリを一意に識別する整数（1 ～ **MAXINT**）を指定します。

empireHistoryCtrlDescr

このエントリによって定義されたデータ収集機能について説明します。

empireHistoryCtrlInterval

エージェントが **MIB** 変数をサンプリングする間隔（秒単位）を示す整数値を指定します。

注: 間隔は、30 秒の倍数にする必要があります。

empireHistoryCtrlObjID

サンプリングする **MIB** 変数の完全なオブジェクト インスタンス識別子を指定します。

注: オブジェクト識別子のインスタンス部分（たとえば、スカラの場合は .0）を含める必要があります。オブジェクトインスタンスが存在している必要があります。Systems Management Empire MIB に含まれている必要があります。

たとえば、**MIB-II**、**Host Resources MIB**、または **Systems Management Empire MIB** 内のサポートされる（整数ベースの）オブジェクトはすべて有効です。オブジェクトは、カウンタ、ゲージ、整数、または列挙整数などの整数タイプにする必要があります。

empireHistoryCtrlObjType

エージェントがサンプリングしている **MIB** 変数の **ASN1/SNMP** タイプを指定します。

empireHistoryCtrlBucketsReq

履歴テーブルに保存する別個のサンプルのリクエストされた数を指定します。利用可能なリソースに応じて、エージェントは、**empireHistoryBucketsGrant** 列をできるだけこの値に近い値に設定します。

empireHistoryCtrlBucketsGrant

このエントリにより定義されるデータ収集機能の履歴テーブルにエージェントが保存する別個のサンプルの実数を指定します。エージェントは、サンプルの最新の **BucketsGrant** 数を保持します。

empireHistoryCtrlLastCall

sysUptime に基づいて、このエントリの代わりにサンプルが取得された最後の時刻を指定します。

empireHistoryCtrlCreateTime

sysUptime に基づいて、この履歴サンプリング機能が作成された時刻を指定します。

empireHistoryCtrlStatus

エントリのステータスを定義します。以下のいずれかになります。

- active
- notInService
- notReady
- createAndGo
- createAndWait
- destroy

これらの値は、SNMPv2 の SMI RowStatus テキスト表記法で定義されます。

ステータスを **destroy(6)** に設定すると、エージェントはこのエントリの履歴サンプリングを中止し、履歴テーブル内のこの行および対応するデータ サンプル行の両方を削除します。

empireHistoryCtrlFlags

ビット フィールドを表します。1つ以上のビットを設定すると、このエントリの動作が変わります。

000 000 80 - エントリは AIM プラグイン (読み取り専用) によって作成されます。

000 001 00 - パフォーマンス キューブ データを収集します。

注: 履歴制御テーブルの詳細については、エージェントのインストールにある **mib** サブディレクトリの仕様 **empire.asn1** と、「**Systems Management Empire MIB**」の章を参照してください。

履歴テーブルの列

以下に、履歴テーブルの列を示します。

注: 履歴テーブルの列の詳細については、MIB の仕様（エージェントのインストールにある `mi` サブディレクトリ内の `empire.asn1`）および「Systems Management Empire MIB」の章を参照してください。

`empireHistoryIndex`

このサンプルが一部となっている履歴制御テーブル内の行を識別します。制御テーブル内の対応するエントリの `empireHistoryCtrlIndex` と同じ値があります。

`empireHistorySampleIndex`

同じ履歴制御エントリと関連付けられたすべてのサンプル間でこれが表す特定のサンプルを一意に識別するインデックスを指定します。このインデックスは 1 から始まり、新しいサンプルが格納されるたびに 1 ずつ増えます。

`empireHistoryStartTime`

`sysUptime` に基づいて、最初のサンプルが取得された時刻を指定します。

`empireHistorySampleTime`

`sysUptime` に基づいて、この特定のサンプルが取得された時刻を指定します。

`empireHistoryValue`

このサンプルで取得された MIB 変数の現在の値を指定します。

行作成オブジェクト

行作成を最適化するには、履歴テーブルと共に以下の MIB オブジェクトを使用できます。

`histCtrlUnusedIndex`

変数で `SNMP Get` を実行するとき、履歴テーブルの未使用のインデックス番号を返します。

権限：読み取り専用

histCtrlMatchDescr

特定のエントリ説明に相当するインデックス番号を決定します。この **MIB** オブジェクトの **SNMP Set** を実行して、エージェントが履歴テーブル内のエントリを検索し、説明が最後に一致したエントリのインデックス値を **histCtrlMatchIndex** MIB オブジェクトに配置するようにします。

権限：読み書き

histCtrlMatchIndex

histCtrlMatchDescr と共に使用すると、インデックス番号を持つ特定のエントリ説明に一致します。

権限：読み取り専用

履歴サンプリングの例

以下の表に、履歴制御テーブル内のサンプルエントリを示します。

インデックス	Desc	Int	ObjID	Type	Bckts-Req	Bckts-Grant	Last-Call	Create-Time	Status
1	1 分 CPU Load Avg.	60	loadAverage1Min.0	2	5	5	0d 6:30:0 0	0d 0:0:0	active
2	Memory In Use	30	memInUse.0	2	10	10	0d 6:30:3 0	0d 0:0:0	active
3	Swap Capacity	900	swapCapacity.0	2	48	48	0d 6:30:0 0	0d 2:00:00	active

履歴制御テーブルは、行を追加および削除するために、**SNMPv2 Structure of Management Information (SMI)** の **RowStatus** テキスト表記法を使用します。

このテーブルでは、行は以下のように定義されます。

- 制御行 1 により、エージェントは 1 分ごとに 1 分間の負荷平均をポーリングし、直前 5 つのサンプル（5 分間）を格納します。
- 制御行 2 により、エージェントは 30 秒ごとに memoryInUse MIB 変数をポーリングし、直前 10 のサンプル（5 分間）を格納します。
- 制御行 3 により、エージェントは 15 分ごとに swapCapacity 変数（使用中のスワップの割合）をポーリングし、直前 48 のサンプルを格納します。48 のサンプルを格納することで、管理システムは 12 時間に一度だけこれらのサンプルをアップロードすればよいことになります（すなわち、15 分間隔で 48 のサンプルのため、12 時間分になります）。

以下の表に、サンプル履歴制御テーブル内で定義されたサンプリング設定の結果履歴テーブル内に格納された履歴サンプルについて説明します。

Index	SampleIndex	StartTime	サンプル時刻	Value
1	387	0d 0:0:0	0d 6:26:00	2
1	388	0d 0:0:0	0d 6:27:00	0
1	389	0d 0:0:0	0d 6:28:00	1
1	390	0d 0:0:0	0d 6:29:00	2
1	391	0d 0:0:0	0d 6:30:00	2
2	772	0d 0:0:0	0d 6:26:00	32204
2	773	0d 0:0:0	0d 6:26:30	32213
:	:	:	:	:
:	:	:	:	:
2	781	0d 0:0:0	0d 6:30:30	32224
3	1	0d 2:0:0	0d 2:00:00	70
3	2	0d 2:0:0	0d 2:15:00	64
3	3	0d 2:0:0	0d 2:30:00	66
:	:	:	:	:
:	:	:	:	:
3	19	0d 2:0:0	0d 6:30:00	80

履歴制御テーブルの使用の詳細については、「履歴の収集」の章を参照してください。

履歴制御テーブルの表示

CA eHealth AdvantEDGE View を使用している場合、システム リストからモニタするシステムを選択し、設定リストから履歴の収集を選択し、設定アイコンをクリックすることにより、履歴情報をシステムにクエリすることができます。

注: 詳細については、CA eHealth AdvantEDGE View Web ヘルプを参照してください。

履歴制御テーブルの設定

履歴制御テーブルでエントリを追加、削除、または変更することで、SystemEDGE エージェントが履歴テーブルに格納する MIB オブジェクトを制御できます。

以下のいずれかの方法で履歴制御テーブルを設定できます。

- 次の 2 つの方法のいずれかを動的に使用します。
 - CA eHealth AdvantEDGE View などの管理システムから SNMP コマンドを使用してテーブルを変更します。
 - CA Virtual Assurance 内でファイル ベースの設定を使用します。

注: 詳細については、「操作時の動的な設定」を参照してください。

SystemEDGE 設定ファイル `sysedge.cf` で履歴制御テーブルのエントリを指定することで、起動時に初期化。

注: 詳細については、「起動時の初期設定」を参照してください。

起動時の初期設定

起動時に、エージェントは `sysedge.cf` 設定ファイルを読み取ります。 `emphistory` ディレクティブを使用して、履歴制御テーブルに該当するエントリを追加することにより、このファイルを使用してエージェントがモニタする MIB オブジェクトを指定できます。

操作時の動的な設定

`emphistory` コマンドラインユーティリティを使用して、履歴制御テーブルのエントリを設定し、履歴制御テーブルから動的にデータサンプルを取得できます。`emphistory` ユーティリティは、エージェントインストールの `bin` サブディレクトリにあります。エージェント設定ファイルに手動で履歴テーブルエントリを追加することもできます。

注: エージェント設定ファイルに履歴テーブルエントリを追加する方法の詳細については、「`emphistory` ユーティリティ -- 履歴制御テーブルでのエントリの管理」を参照してください。

`emphistory` ディレクティブ -- 履歴制御テーブルへのエントリの追加

`emphistory` ディレクティブでは、履歴制御テーブルにエントリを追加できます。`sysedge.cf` ファイルで履歴制御テーブル内の列を表す、このディレクティブとその引数を使用して、履歴サンプリング用に収集する MIB オブジェクトを定義します。

履歴制御テーブルにエントリを追加するには、以下のように `sysedge.cf` ファイルで `emphistory` ディレクティブを使用します。

```
emphistory index interval objid buckets 'description' flags
index
```

エージェントがエントリを作成する行番号を指定します。

```
interval
```

エージェントがオブジェクトの値をサンプリングする間隔（秒単位）を指定します。間隔は、30 秒の倍数にする必要があります。

```
objid
```

値をサンプリングするエージェントの MIB の内のオブジェクトインスタンスを指定します。ドットで区切った完全な 10 進数値（たとえば、`1.3.6.1.2.1.25.1.5.0` など）またはシンボリック MIB 名（たとえば、`hrSystemNumUsers.0` など）を使用して、OID を指定できます。どちらの場合も、オブジェクトインスタンスを指定する必要があります。これは、通常表形式以外の MIB 変数の場合はゼロです。

buckets

エージェントが内部で格納するバケット（すなわちサンプル）の数を指定します。エージェントは、サンプルの最後のバケット番号を格納します。エージェントが新しいサンプルを取得するたびに、最も古いサンプルが削除されます。

'description'

このエントリの説明フィールドを示す、引用符付きの文字列（0 ～ 128 文字の長さ）を指定します。

empireHistoryCtrlFlags

ビットフィールドを表します。1つ以上のビットを設定すると、このエントリの動作が変わります。

000 000 80 - エントリは AIM プラグイン（読み取り専用）によって作成されます。

000 001 00 - パフォーマンス キューブ データを収集します。

flags

ビットフィールドを表します。1つ以上のビットを設定すると、このエントリの動作が変わります。

000 000 80 - エントリは AIM プラグイン（読み取り専用）によって作成されます。

000 001 00 - パフォーマンス キューブ データを収集します。

履歴の収集の例

以下のエントリは、120 秒ごとにテーブル インデックス 15 を使用してオブジェクト MemInUse の値をサンプリングし、最後の 60 のサンプルを格納するようにエージェントに指示します。

```
emphistory 15 120 memInUse.0 60 'MemInUse history'
```

履歴制御テーブルのエントリを管理するための `emphistory` コマンド

`emphistory` コマンドは、コマンドの利用可能なリスト (`add`、`setstatus`、`delete`、`dump`、および `list`) から実行するコマンドのタイプを指定します。

コマンドのタイプによっては、コマンドを完了するために追加のパラメータを指定する必要があります。コマンドの構文は、以下のとおりです。

```
add index objid interval buckets "description" flags
```

```
setstatus index status
```

```
delete index
```

```
dump index
```

```
list
```

```
index
```

エージェントの履歴制御テーブルでこのエントリの行番号を指定します。

行のインデックスは **1** から始まります。ダンプ操作を指定すると、エージェントはインデックスにより識別される履歴制御テーブルのエントリに対応する履歴テーブルからすべてのデータ サンプルを取得します。ダンプ操作でインデックス値 **-1** を指定した場合、エージェントは履歴テーブルの内容をすべて取得します。

行作成で使用する未使用の行の特定の **MIB** テーブルインデックスを指定するには、追加操作のインデックス値を指定する必要があります。

`objid`

エージェントがサンプリングして履歴制御テーブルに格納する **MIB** 変数の完全なオブジェクトインスタンス識別子を指定します。ドットで区切った完全な **10** 進数値 (たとえば、**1.3.6.1.2.1.25.1.5.0** など) またはシンボリック **MIB** 名 (たとえば、**hrSystemNumUsers.0** など) を使用して、**OID** を指定できます。**OID** (つまり、スカラの場合は **0**) のインスタンス部分を提供する必要があります。オブジェクトインスタンスは存在している必要があり、整数タイプ (カウンタ、ゲージ、整数、または列挙整数が含まれます) でなければなりません。

interval

連続するサンプルの間隔を秒単位で指定します。この値は、**30 秒**の倍数にする必要があります。

buckets

この制御エントリの代わりに履歴テーブルに保存される不連続データサンプルの数を指定します。バケットと呼ばれる各サンプルには、**MIB** 変数のスナップショット値、サンプルが作成された時間、サンプルインデックス、およびデータ収集機能を定義する履歴制御テーブルのエントリに相当するインデックスが含まれています。履歴テーブルには、サンプルの最新のバケット番号が格納されます。

status

setstatus 操作と共に使用されると、履歴制御テーブルで行のステータスを設定する際に使用する、**RowStatus** テキスト表記法の値を指定します。**status** パラメータは、以下のいずれかの値になります。

- (1)active
- (2)notInService
- (6)destroy

ステータスを **destroy(6)** に設定すると、エージェントはそのエントリの履歴サンプリングを中止し、履歴制御テーブル内の行および履歴テーブル内の対応するデータ サンプル行を削除します。

'description'

(オプション)。行の説明を指定します。値を指定した場合、**emphistory** はデフォルトの履歴制御エントリ説明文字列の代わりに、指定された文字列を使用します。説明を含めた場合、一重引用符 (") で囲む必要があります。説明は、引用符を含めずに **128** 文字未満にする必要があります。それより長い文字列は切り捨てられます。

フラグ

ビット フィールドを表します。1つ以上のビットを設定すると、このエントリの動作が変わります。

000 000 80 - エントリは AIM プラグイン (読み取り専用) によって作成されます。

000 001 00 - パフォーマンス キューブ データを収集します。

重要: `emphistory` ユーティリティを以下の方法で使用することはできなくなる予定です。以下の形式は今後サポートされなくなるため、新しい引数を使用することが強く推奨されます。

```
emphistory add ipaddr[:port] commstr ctrlIndex objid interval buckets ['description']
```

emphistory の例

このセクションでは、SNMP バージョン 1 および 3 で `emphistory` ユーティリティを使用する例を示します。エージェントは、デフォルト ポート 161 で実行されていると仮定します。

例: 履歴制御テーブルのエントリのリスト

以下の例では、エージェントの履歴制御テーブルの内容がリストされます。

```
emphistory -h 127.0.0.1 -v 1 -c public -o list
```

```
emphistory -h 127.0.0.1 -v 3 -u username -s 3 -A MD5 -a authPassword -X DES -x encryptPassword -o list
```

例: 履歴制御エントリの追加

以下の例では、エージェントの履歴制御テーブルに、テーブル インデックス 5 で新しい制御エントリを追加します。この制御エントリは、60 秒ごとに `ifInOctets.1` MIB オブジェクト インスタンスをサンプリングし、直前 10 のサンプルを格納するようにエージェントに指示します。

```
emphistory -h 127.0.0.1 -v 1 -c private -o add 5 ifInOctets.1 60 10
```

```
emphistory -h 127.0.0.1 -v 3 -u username -s 3 -A MD5 -a authPassword -X DES -x encryptPassword -o add 5 ifInOctets.1 60 10
```

例: 履歴制御エントリの削除

以下のエントリでは、テーブルインデックス 3 で履歴制御テーブルエントリを削除します。

```
emphistory -h 127.0.0.1 -v 1 -c private -o delete 3
```

```
emphistory -h 127.0.0.1 -v 3 -u username -s 3 -A MD5 -a authPassword -X DES -x  
encryptPassword -o setstatus 3 destroy
```

注: これらのエントリは、このコントロールエントリに対応する履歴テーブル内の格納されたデータ サンプルを削除するようにもエージェントに指示します。

例: 制御エントリの行ステータスの設定

以下の例では、テーブルインデックス 5 で履歴制御テーブルの制御エントリを無効にしますが、対応する格納されたサンプルを履歴テーブルに保存します。

```
emphistory -h 127.0.0.1 -v 1 -c private -o setstatus 5 2
```

```
emphistory -h 127.0.0.1 -v 3 -u username -s 3 -A MD5 -a authPassword -X DES -x  
encryptPassword -o setstatus 5 2
```

2

RowStatus テキスト表記法の値 **notInService(2)** に対応します。

例: 格納されたデータ サンプルの取得

以下の例では、履歴制御テーブルの行 5 で定義されたデータ収集機能に対応する、格納されたデータ サンプルをすべて取得します。

```
emphistory -h 127.0.0.1 -v 1 -c private -o dump 5
```

```
emphistory -h 127.0.0.1 -v 3 -u username -s 3 -A MD5 -a authPassword -X DES -x  
encryptPassword -o dump 5
```

以下の例では、-1 を使用してすべてのコントロールエントリの格納されたサンプルをすべて取得します。このコマンドは、履歴テーブルの全内容を取得します。

```
emphistory -h 127.0.0.1 -v 1 -c private -o dump -1
```

```
emphistory -h 127.0.0.1 -v 3 -u username -s 3 -A MD5 -a authPassword -X DES -x  
encryptPassword -o dump -1
```

第 15 章: カスタム MIB オブジェクト

この章では、カスタム MIB オブジェクトを追加することで、Systems Management Empire MIB を拡張する方法について説明します。

このセクションには、以下のトピックが含まれています。

[Systems Management Empire MIB 拡張グループ](#) (P. 467)

[拡張グループ機能](#) (P. 468)

[拡張変数の設定](#) (P. 469)

[拡張の例](#) (P. 471)

[拡張スクリプト](#) (P. 473)

[管理ソフトウェアでの拡張変数の使用](#) (P. 475)

[拡張の使用に関する推奨事項](#) (P. 477)

Systems Management Empire MIB 拡張グループ

SystemEDGE エージェントには、システム管理 MIB を拡張してシステムとアプリケーションに関する情報を含めるためのメカニズムが用意されています。この機能を使用すると、エージェントを拡張してシステム環境をより効率的に管理できます。アプリケーション固有の MIB 変数を設計することで、アプリケーションソース内で SNMP サポートを実装せずに SNMP を使用してアプリケーションを管理することもできます。

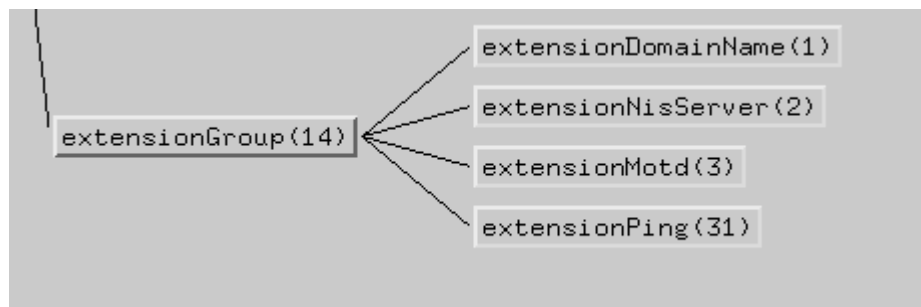
SystemEDGE エージェントでは、システム管理 MIB の拡張グループ

(`extensionGroup`) によってこの拡張がサポートされます。このグループには、設定可能な未指定のスカラ MIB 変数が含まれています。これらの変数のいずれかの SNMP `GetRequest` に応じて、SystemEDGE エージェントは、変数に指定したコマンドを呼び出し、コマンドが返す値を返します。SNMP `Set` 操作を使用すると、パラメータをコマンドへ渡すこともできます。

拡張グループは、UNIX および Windows システムでサポートされます。Windows では、Windows レジストリで利用可能なパフォーマンスおよび設定データを報告するようにエージェントを設定することもできます。このレポートをサポートするため、SystemEDGE エージェントには Windows レジストリ拡張グループも用意されています。

注: このグループの詳細については、「Windows レジストリとパフォーマンス MIB オブジェクト」の章を参照してください。

以下の図に、エージェントと共に分散した 4 つのサンプル拡張変数を示します。



拡張グループ機能

システム管理 MIB の拡張グループは、OID 1.3.6.1.4.1.546.14 に存在します。このグループには、2 ユイのユーザ定義スカラ MIB 変数の領域が用意されています。これらの新しい変数は 1 ～ 2 ユイの順で番号が付けられ、他のスカラ MIB オブジェクトのように参照されます。たとえば、拡張オブジェクト 1 は **Get** および **Set** リクエストメッセージで 1.3.6.1.4.1.546.14.1.0 として参照されます。どの場合でも、拡張変数はオブジェクトインスタンス識別子 .0 を使用します。

変数は、読み取り専用または読み書きとして指定できます。拡張変数は、以下のような任意の有効なベース **SNMP** タイプになります。

- Integer
- Counter
- Gauge
- Octetstring
- TimeTicks
- ObjectId
- IpAddress

注: デバッグ オプションをオンにしてエージェントを開始する詳細については、「エージェントの開始」の章を参照してください。

拡張変数の設定

拡張変数は、SystemEDGE エージェントの設定ファイル `sysedge.cf` で設定できます。起動時、エージェントはこれらの値を設定し、変数に関連付けられたプログラムまたはコマンドが実行可能かどうかを確認します。設定ファイル内では、キーワード拡張により拡張変数が定義されます。次のセクションでは、このキーワードを使用する方法について説明します。

拡張キーワード -- 拡張グループへのエントリの追加

拡張キーワードを使用して、Systems Management Empire MIB 拡張グループにエントリを追加できます。モニタに利用できるようにするカスタム MIB オブジェクトを指定するには、sysedge.cf ファイルでキーワードを使用します。

以下のように、拡張キーワードを使用して拡張グループにエントリを追加します。

```
extension LeafNumber Type Access 'Command'
```

LeafNumber

このエントリによって定義された 1 ～ 2³² の範囲で拡張変数番号を指定します。

Type

このエントリの SNMP タイプを指定します。サポートされるタイプは、以下のとおりです。

- Integer
- Counter
- Gauge
- Octetstring
- TimeTicks
- Objectid
- IPAddress

Access

アクセス タイプを指定します。読み取り専用または読み書きのいずれかです。

'Command'

実行するコマンドのフルパスを指定する（パラメータがある場合はパラメータ付きで）、引用符で囲まれた文字列を 0 ～ 256 文字の長さで指定します。このコマンドは、プラットフォームに依存しない拡張名にすることも、**Get**、**GetNext**、または **Set** リクエストのいずれかを介してこの変数にアクセスしたときに実行されるコマンドのフルパスにすることもできます。プラットフォームに依存しない拡張の場合、**<ExtensionName>** という名前の実行可能ファイルが **CASYSEDGE/extension/<ExtensionName>** で検索されます。コマンドファイルに現在アクセスできない場合、この変数の設定に失敗します。

追加パラメータ

拡張コマンドの一部としての設定ファイルで指定したパラメータに加えて、SystemEDGE エージェントはリクエストの処理方法の決定に役立ついくつかの追加パラメータを渡します。これらのパラメータは、拡張ステートメントの一部として引用符で指定したパラメータの後に渡されます。

以下のリストでは、拡張コマンドに渡すことができる追加パラメータについて説明します。

Leafnumber

この変数のリーフ番号（1 ～ 232）を表す整数値を指定します。複数の拡張値をサポートする 1 つのスクリプトがある場合、どこのパラメータを使用してリクエストする変数を決定できます。

Request-Type

SNMP リクエストのタイプを示します。以下のいずれかです。

- Get
- GetNext
- Set

リクエストタイプは、すべての文字で渡されます。

Set-Value

Set リクエストで渡された値が含まれる文字列を指定します。後で **Get** または **GetNext** がこの値を返すことができるような方法で変数の現在値を変更するには、拡張プログラムでこの文字列を使用します。

拡張の例

SystemEDGE エージェントには、いくつかのサンプル拡張変数が含まれています。これらの例は、サンプルの `sysedge.cf` ファイルに定義されています。これらの例を実装したスクリプトが SystemEDGE エージェントのインストールの `contrib` サブディレクトリに含まれています。これらの変数は、Systems Management Empire MIB（エージェントインストールの `mib` サブディレクトリの `empire.asn1`）にも定義されています。

重要: 独自の拡張を追加する前に、この章に示す例や Systems Management Empire MIB 内の例を注意深く確認してください。わかりやすくするため、これらの例には適切な設定ファイルの拡張コマンドが含まれています。

これらの拡張を `sysedge.cf` ファイルに追加して、SystemEDGE エージェントで利用可能にすることができます。

例: プラットフォーム非依存の拡張

以下の拡張オブジェクトは、コマンド
`/opt/CA/SystemEDGE/extension/myextension/myextension` (Unix の場合) または
「`C:¥Program Files¥CA¥SystemEDGE¥extesnion¥myextension¥myextension.<EXT>`」
(Windows の場合) の出力を返します (<EXT> は、環境変数 `%PATHEXT%` から実行可能な拡張のいずれかです)。

```
extension 1 OctetString ReadOnly myextension
```

このオブジェクトのインスタンス ID は 1.3.6.1.4.1.546.14.1.0 です。

例: リモート ping (UNIX および Windows)

エージェントが実行されているホストからリモートホストに対して `ping` を実行するように SystemEDGE エージェントに指示するには、以下の拡張オブジェクトを使用します。

UNIX システムでは、`sysedge.cf` ファイルに以下を入力します。

```
extension 31 OctetString ReadOnly "/bin/ping www.ca.com"
```

Windows システムでは、`%SystemRoot%¥System32¥sysedge.cf` ファイルに以下を入力します。

```
extension 31 OctetString ReadOnly "%WINDIR%¥system32¥ping.exe www.ca.com"
```

このオブジェクトのインスタンス ID は、どちらの例でも 1.3.6.1.4.1.546.14.31.0 です。

拡張スクリプト

SystemEDGE エージェントでは、拡張スクリプトの操作の制限がほとんどありません。ただし、以下の条件があります。

- すべての操作（Set、Get、または GetNext）に出力が必要です。Set 呼び出しの出力では、実際に設定された値がエコーされますが、Get と GetNext の出力はオブジェクトの値になります。出力がない場合、SNMP クエリは失敗します。
- 拡張スクリプトからの出力は、最初の改行文字までのみ解析されます。最初の文字が改行の場合、出力は NULL と見なされ、SNMP クエリが失敗してエラーが返されます。

注: SystemEDGE エージェントは root または管理者として実行されるため、すべてのコマンドおよびスクリプトで絶対パス名が使用されていて、完全にデバッグされており、あいまいなコードや不必要なオプションが含まれていないことを確認してください。

Windows システムでは、拡張スクリプトの書き込みにバッチ ファイルを使用できます。エージェントはそれらのバッチ ファイルを直接実行できます。ただし、バッチ ファイル機能は大幅に制限されます。Windows では、代わりに Perl および他のスクリプト言語を使用してください。

サンプル スクリプトのテスト

拡張スクリプトを作成したら、コマンドラインでテストします。以下の例では、UNIX システム上で **OctetString** 拡張を作成し、その出力をテストし、次に **SNMP** を使用してスクリプトから値を返します。

スクリプトをテストする方法

1. 以下の情報を含む、`/opt/CA/SystemEDGE/debugext.sh` という名前の実行可能ファイルを作成します。

```
#!/bin/sh
Script=$0
LeafNumber=$1
Operation=$2
SetValue=$3

if [ "$Operation" = "SET" ]; then
echo $Script      > /tmp/debugext_set.txt
echo `date`      >> /tmp/debugext_set.txt
echo $LeafNumber >> /tmp/debugext_set.txt
echo $Operation  >> /tmp/debugext_set.txt
echo $SetValue   >> /tmp/debugext_set.txt
echo $SetValue   > /tmp/debugext_setvalue.txt #SET VALUE HERE
echo $SetValue   #You must echo back what was set
fi

# Override GETNEXT and just do GET
if [ "$Operation" = "GETNEXT" ]; then
Operation="GET"
fi

if [ "$Operation" = "GET" ]; then
echo $Script      > /tmp/debugext_get.txt
echo `date`      >> /tmp/debugext_get.txt
echo $LeafNumber >> /tmp/debugext_get.txt
echo $Operation  >> /tmp/debugext_get.txt
if [ ! -f /tmp/debugext_setvalue.txt ]; then
echo "ERROR: do set first... Script:$Script
LeafNumber:$LeafNumber Operation:$Operation"
exit 0
fi
cat /tmp/debugext_setvalue.txt #return SET VALUE
fi
```

注: この例では、**OctetString** である拡張スクリプトをテストします。UNIX オペレーティングシステムで有効です。Windows システムでは、アクションスクリプト コマンドでインタープリタをコールする必要があります。たとえば、「perl.exe myscript.pl」と入力します。

2. sysedge.cf に以下の行を追加します。

```
extension 1 OctetString ReadWrite /opt/CA/SystemEDGE/debugext.sh
```

注: この例では、OctetString である拡張スクリプトをテストします。UNIX オペレーティングシステムで有効です。Windows システムでは、アクション スクリプト コマンドでインタープリタをコールする必要があります。たとえば、「perl.exe myscript.pl」と入力します。

3. コマンドラインで以下の内容を入力して、myset.txt という名前のファイルを作成します。

```
echo "1.3.6.1.4.1.546.14.1.0 04 debugSetString" > myset.txt
```

これで、myset.txt という名前のファイルが作成されます。OctetString (04) "debugSetString" を OID 1.3.6.1.4.1.546.14.1.0 に設定します。

4. 以下の内容を入力して、SNMP Set を発行します。

```
./snmpset -c private -h 127.0.0.1 < myset.txt
```

5. 以下の内容を入力して、debugSetString の値を取得します。

```
./snmpget -c public -h 127.0.0.1 -o 1.3.6.1.4.1.546.14.1.0
```

6. コマンドラインで以下の内容を入力することで、セットアップをテストします。

```
./debugext.sh 1 SET debugSetString2
```

このコマンドは、出力として設定されている値を返します。

```
./debugext.sh 1 GET
```

このコマンドは、以前の SET コールで設定された値を返します。

管理ソフトウェアでの拡張変数の使用

管理システム ソフトウェアに拡張変数を組み込むどの方法でも、MIB 仕様ファイルを編集してインポートする必要があります。このガイドでは、MIB 仕様ファイルを管理システム ソフトウェアにインポートする方法の詳細については説明しませんが、拡張変数を組み込むための全体的な 2 つの戦略について説明します。

- empire.asn1 を編集して、サイトに定義された拡張変数を含めます。
- 別個の MIB ファイルを編集して、サイトに定義された拡張変数を含めます。

empire.asn1 で拡張変数に関する編集を行う方法

以下の手順に従って、独自の拡張変数をシステム管理 MIB (`empire.asn1`) に追加します。

1. 関連する拡張スクリプトを作成およびデバッグし、次にエージェントの設定ファイル (`sysedge.cf`) でそれらを設定して含めます。
2. `empire.asn1` を編集して、`extensionGroup` の下に存在する新しい拡張 MIB 変数を含めます。
3. `empire.asn1` のテスト コンパイルを実行して、構文エラーがないことを確認します。この手順は、管理システムと MIB コンパイラに固有です。
4. 新しい `empire.asn1` ファイルを管理システム ソフトウェアにインポートします。この手順は、管理システムと MIB コンパイラに固有です。

注: ユーザが MIB ファイルを再インポートしない場合、管理システム ソフトウェアは新しい拡張 MIB オブジェクトにアクセスできません。

別個の MIB 仕様で拡張変数に関する編集を行う方法

更新しやすいように、拡張変数を別個の MIB 仕様ファイルに置くことができます。`empire.asn1` よりかはるかに小さい MIB 仕様ファイルに変更を加えると、`empire.asn1` ファイル全体を再コンパイルおよび再びロードする必要がないため、時間と労力を節約できます。

拡張変数を別個の MIB 仕様ファイルに追加する方法

1. 関連する拡張スクリプトを作成およびデバッグし、エージェントの設定ファイル (`sysedge.cf`) でそれらを設定して含めます。
2. 独自の MIB 仕様ファイル（たとえば、`extensions.asn1` など）を編集して、システム管理拡張グループの下に存在する新しい拡張 MIB 変数を含めます。
3. `extensions.asn1` のテスト コンパイルを実行して、構文エラーがないことを確認します。この手順は、管理システムと MIB コンパイラに固有です。
4. `extensions.asn1` ファイルを管理システム ソフトウェアにインポートします。この手順は、管理システムと MIB コンパイラに固有です。

注: ユーザが MIB ファイルを再インポートしない場合、管理システム ソフトウェアは新しい拡張 MIB オブジェクトにアクセスできません。

拡張の使用に関する推奨事項

SystemEDGE 拡張を作成するときは、以下のガイドラインに従います。

- `empire.asn1` は編集しないでください。別個の MIB 仕様を使用します。
- 出力、プラグイン、および拡張の問題を見つけるには、SystemEDGE デバッグ（ログ ファイル モニタリング）を使用します。
- コマンドラインの長さに関する Windows の制限を満たすため、拡張エントリは 128 文字以下にします。長さには、パス、コマンド、および引数が含まれる点に注意してください。
- バッチ ファイルは使用しないでください。代わりに、Windows スクリプト、VBScript、Perl、C、シェルなどを使用します。
- 新しい拡張の `fork/exec` を監視します。拡張は同期的に使用されるため、他のアクションをブロックする可能性があります。

第 16 章: Windows レジストリとパフォーマンス MIB オブジェクト

この章では、Windows レジストリとパフォーマンス データ用のオブジェクトを Systems Management Empire MIB に追加する方法について説明します。

このセクションには、以下のトピックが含まれています。

[Windows レジストリおよびパフォーマンス機能 \(P. 479\)](#)

[Systems Management Empire MIB の ntRegPerf グループ \(P. 480\)](#)

[Windows レジストリおよびパフォーマンス変数の設定 \(P. 483\)](#)

[Windows レジストリとパフォーマンスの例 \(P. 485\)](#)

[管理ソフトウェアでの Windows レジストリおよびパフォーマンス変数の使用 \(P. 486\)](#)

Windows レジストリおよびパフォーマンス機能

SystemEDGE には、Windows レジストリおよびパフォーマンス カウンタからシステム管理 MIB と Systems Management Empire MIB を拡張するための強力なメカニズムが用意されています。この情報には、設定データ（通常は Windows プログラム `regedit` を使用して表示します）およびパフォーマンス データ（通常は Windows プログラム `perfmon` を使用して表示します）が含まれます。

この機能を使用すると、システムとアプリケーションの追加の設定およびパフォーマンス データを返すようにエージェントをカスタマイズできます。たとえば、エージェントを使用して、SNMP を介して多くのアプリケーション レジストリ エントリを利用できるようにすることができます。さらに、多くのアプリケーションのパフォーマンス統計にアクセスできます。

Systems Management Empire MIB の ntRegPerf グループ

SystemEDGE エージェントでは、Systems Management Empire MIB の ntRegPerf グループの追加の Windows レジストリおよびパフォーマンス パラメータがサポートされます。Systems Management Empire MIB の ntRegPerf グループは OID 1.3.6.1.4.1.546.5.7 に存在します。このグループには、最大 128 のユーザ定義スカラ MIB 変数の領域が用意されています。これらの新しい変数は 1 ~ 128 の順で番号が付けられ、他のスカラ MIB オブジェクトと同じように参照されます。たとえば、ntRegPerf オブジェクト 1 は Get リクエストメッセージで 1.3.6.1.4.1.546.5.7.1.0 として参照されます。どの場合でも、ntRegPerf 変数はオブジェクトインスタンス識別子 .0 を使用します。これらの変数のいずれかの SNMP Get リクエストに応じて、エージェントは Windows レジストリを読み取り、取得された値を返します。

ntRegPerf 変数は、以下のような任意の有効なベース SNMP タイプになります。

- Integer
- Counter
- Gauge
- OctetString
- TimeTicks
- ObjectId
- IpAddress

レジストリ データ

SystemEDGE には、標準設定レジストリのデータが用意されています。このデータは、キー名および値の両方によってインデックスが付けられます。たとえば、キー SYSTEM¥CurrentControlSet¥Control¥CrashControl および値 DumpFile により、システム ダンプ ファイルの場所を説明するテキスト文字列が特定されます。LOCAL_MACHINE レジストリ データのみがサポートされます。

以下のリストに、SystemEDGE でサポートされるレジストリ データ タイプと推奨される SNMP タイプの対応を示します。

REG_DWORD

Integer

REG_SZ

OctetString

REG_EXPAND_SZ

OctetString

REG_MULT_SZ

OctetString (最初の文字列だけが返されます)

パフォーマンス データ

SystemEDGE では、オブジェクトとカウンタ名を指定することによりパフォーマンス レジストリのパフォーマンス カウンタにアクセスできます。たとえば、オブジェクト NWLink NetBIOS とカウンタ「Bytes Total/sec」は NetBIOS のバイト カウンタ統計を識別します。

パフォーマンス カウンタを使用するときは、データを慎重に解釈する必要があります。Windows には少なくとも 27 の異なるカウンタ タイプがあります。ほとんどのカウンタは、実際にカウンタの名前から推測される形式でデータを報告するわけではありません。むしろ、ほとんどの値は後処理（たとえば、2つのサンプルの差を経過時間で除算するなど）が必要な生のカウンタです。「Bytes Total/sec」のようなカウンタ名が表示された場合、実際に取得されるデータは、このレートの計算に使用できる生のカウンタです。

この機能を使用するには、Windows パフォーマンス カウンタに慣れておいてください。詳細については、Windows リソース キット内の Windows の最適化についてのセクションを参照してください。

以下のリストに、SystemEDGE でサポートされる 4 バイト パフォーマンス データタイプと推奨される SNMP タイプの対応を示します。

PERF_COUNTER_COUNTER

カウンタまたは整数

PERF_COUNTER_RAWCOUNT

カウンタ、整数、またはゲージ

PERF_COUNTER_RAWCOUNT_HEX

カウンタ、整数、またはゲージ

PERF_SAMPLE_COUNTER

カウンタ、整数、またはゲージ

以下のリストに、SystemEDGE でサポートされる 8 バイト パフォーマンス データタイプと推奨される SNMP タイプの対応を示します。SNMPv1 では 4 バイト値のみサポートされるため、エージェントは最小桁数である 4 バイトのデータのみ返します。

PERF_COUNTER_TIMER

カウンタ、整数、またはゲージ

PERF_COUNTER_BULK_COUNT

カウンタ、整数、またはゲージ

PERF_COUNTER_LARGE_RAWCOUNT

カウンタ、整数、またはゲージ

PERF_COUNTER_LARGE_RAWCOUNT_HEX

カウンタ、整数、またはゲージ

PERF_COUNTER_TIMER_INV

カウンタ、整数、またはゲージ

PERF_AVERAGE_BULK

カウンタ、整数、またはゲージ

PERF_100NSEC_TIMER

カウンタまたは整数

PERF_100NSEC_TIMER_INV

カウンタまたは整数

PERF_COUNTER_MULTI_TIMER

カウンタまたは整数

PERF_COUNTER_MULTI_TIMER_INV

カウンタまたは整数

PERF_100NSEC_MULTI_TIMER

カウンタまたは整数

PERF_100NSEC_MULTI_TIMER_INV

カウンタ、整数、またはゲージ

PERF_ELAPSED_TIME

整数

PERF_RAW_FRACTION

整数またはゲージ

サポートされないパフォーマンス データタイプ

複数のサンプルおよび内部データが含まれる複数のカウンタは、1つの値で示すことが難しく、かなりの後処理が必要です。このため、CA SystemEDGE では以下のカウンタ タイプがサポートされません。

- PERF_COUNTER_QUEUELEN_TYPE
- PERF_COUNTER_TEXT
- PERF_COUNTER_NODATA
- PERF_SAMPLE_BASE
- PERF_AVERAGE_TIMER
- PERF_AVERAGE_BASE
- PERF_COUNTER_MULTI_BASE
- PERF_RAW_BASE
- PERF_COUNTER_HISTOGRAM_TYPE

Windows レジストリおよびパフォーマンス変数の設定

Windows レジストリとパフォーマンス変数は、`sysedge.cf` ファイルで設定します。SystemEDGE が起動すると、これらの値が設定され、各変数に関連付けられた値にアクセスできることが確認されます。アクセスできない場合、`sysedge.log` にエラーメッセージが出力され、MIB オブジェクトは作成されません。

ntregperf キーワード --Windows レジストリとパフォーマンス MIB オブジェクトの追加

sysedge.cf ファイルで ntRegPerf グループの Windows レジストリとパフォーマンス MIB オブジェクトを定義するには、ntregperf キーワードを使用します。

以下のように、ntregperf キーワードを使用して ntRegPerf グループのエントリを追加できます。

レジストリ データ :

```
ntregperf LeafNumber Type Registry 'Key' 'Value'
```

パフォーマンス データ :

```
ntregperf LeafNumber Type Performance 'Object' 'Counter' 'PerfInstance'
```

LeafNumber

ntRegPerf 変数番号を **1 ~ 128** の範囲で定義します。

Type

このエントリの **SNMP** タイプを指定します。以下のいずれかです。

- Integer
- Counter
- Gauge
- Octetstring
- TimeTicks
- Objectid
- IPAddress

Registry

設定レジストリ エントリを選択します。

Performance

設定パフォーマンス エントリを選択します。

'Key'

アクセスするレジストリ キーを指定する、引用符付きの文字列を **0 ~ 512** 文字の長さで指定します。

'Value'

アクセスするレジストリ キーを指定する、引用符付きの文字列を **0** ～ **128** 文字の長さで指定します。

'Object'

アクセスするパフォーマンス オブジェクトを指定する、引用符付きの文字列を **0** ～ **512** 文字の長さで指定します。

'Counter'

アクセスするオブジェクトのパフォーマンス カウンタを指定する、引用符付きの文字列を **0** ～ **128** 文字の長さで指定します。

'PerfInstance'

アクセスするパフォーマンス カウンタ インスタンスを指定します。それは **Windows perfmon** ユーティリティでリスされるインスタンスと同じになるはずです。

Windows レジストリとパフォーマンスの例

SystemEDGE エージェントには、いくつかのサンプル ntRegPerf 変数が用意されています。これらの例は、サンプル sysedge.cf ファイルで定義されています。独自の ntRegPerf 拡張を追加する前に、Systems Management Empire MIB（エージェントのインストールの mib サブディレクトリ内の empire.asn1）でこれらの例とその定義について調べてください。明確にするため、これらの例には該当する設定ファイル ntRegPerf コマンドが含まれています。

例: CrashControl DumpFile

以下の ntRegPerf オブジェクトは、ダンプ ファイルのパスを返します。

```
ntregperf 1 OctetString Registry 'SYSTEM%CurrentControlSet%Control%CrashControl'  
'DumpFile'
```

このオブジェクトのオブジェクト インスタンス識別子は **1.3.6.1.4.1.546.5.7.1.0** です。

例:スレッドの総数

以下の ntRegPerf オブジェクトは、システムで現在利用可能なスレッドの総数を返します。

```
ntregperf 2 Gauge Performance 'Objects' 'Threads' '1'
```

このオブジェクトのオブジェクト インスタンス識別子は 1.3.6.1.4.1.546.5.7.2.0 です。

例:送信 TCP セグメント数/秒

以下の ntRegPerf オブジェクトは、システムによって送信された TCP セグメントの総数を返します。

```
ntregperf 3 Counter Performance 'TCP' 'Segments Sent/sec' '1'
```

このオブジェクトのオブジェクト インスタンス識別子は 1.3.6.1.4.1.546.5.7.3.0 です。

管理ソフトウェアでの Windows レジストリおよびパフォーマンス変数の使用

Windows レジストリおよびパフォーマンス変数を管理システム ソフトウェアに組み込む方法はいくつかあり、どの方法でも MIB 仕様ファイルを編集してインポートする必要があります。このガイドでは、MIB 仕様ファイルを管理システムソフトウェアにインポートする方法の詳細については説明しませんが、Windows レジストリとパフォーマンス変数を組み込むための全体的な 2 つの戦略について説明します。

- empire.asn1 を編集して、サイトに定義された変数を含めます。
- 別個の MIB ファイルを編集して、サイトに定義された変数を含めます。

empire.asn1 で ntRegPerf 変数に関する編集を行う方法

以下のプロセスを使用して、独自の ntRegPerf 変数を Systems Management Empire MIB に追加できます。

1. sysedge.cf ファイルで、関連する ntRegPerf エントリを作成およびデバッグして、それらを含めます。
2. empire.asn1 を編集して、新しい ntRegPerf MIB 変数を含めます。

3. `empire.asn1` のテスト コンパイルを実行して、構文エラーがないことを確認します。この手順は、管理システムとそれに対応する MIB コンパイラに固有です。
4. 新しい `empire.asn1` ファイルを管理システム ソフトウェアにインポートします。この手順は、管理システムと MIB コンパイラに固有です。

注: ユーザが MIB ファイルを再インポートしない場合、管理システム ソフトウェアは新しい MIB オブジェクトにアクセスできません。

ntRegPerf 変数に関する別個の MIB 仕様を追加する方法

MIB ファイルを直接操作しなくてもよいように、別個の MIB 仕様ファイルに `ntRegPerf` 変数を置くことができます。 `empire.asn1` よりはるかに小さい MIB 仕様ファイルに変更を加えることで、時間と労力を節約できます。この場合、 `empire.asn1` ファイル全体を再コンパイルおよび再びロードする必要があります。

`ntRegPerf` 変数に関する別個の MIB 仕様を追加するには、以下のプロセスを実行します。

1. 独自の MIB 仕様ファイル (たとえば、`ntregperf.asn1` など) を編集して、**Systems Management Empire MIB** の `ntregperfGroup` の下に存在する新しい `ntRegPerf` MIB 変数を含めます。
2. `ntregperf.asn1` ファイルのテスト コンパイルを実行して、構文エラーがないことを確認します。この手順は、管理システムとそれに対応する MIB コンパイラに固有です。
3. `ntregperf.asn1` ファイルを管理システム ソフトウェアにインポートします。この手順は、ネットワーク管理ステーションとそれに対応する MIB コンパイラに固有です。

注: ユーザが MIB ファイルを再インポートしない場合、管理システム ソフトウェアは新しい MIB オブジェクトにアクセスできません。

第 17 章: SystemEDGE AIM

AIM は、ベース SystemEDGE にモニタ機能を追加するプラグイン モジュールです。この章では、エージェントに付属する AIM の設定について説明します。AIM アーキテクチャの詳細については、「概念」の章を参照してください。

このセクションには、以下のトピックが含まれています。

[コア AIM](#) (P. 489)

[トップ プロセス AIM](#) (P. 489)

[モニタ保守ウィンドウ AIM](#) (P. 490)

[パフォーマンス キューブ AIM](#) (P. 491)

コア AIM

monwin と perfcube の AIM は、SystemEDGE コア AIM です。コア AIM は、SystemEDGE コア パッケージと共にインストールされます。これらの AIM には、sysedge.cf ファイル内の sysedge_plugin ステートメントが必要ありません。この機能が必要な場合、SystemEDGE は SystemEDGE コア AIM ライブラリを自動的にロードします。

- monwin エントリが sysedge.cf ファイル内に見つかる場合、エージェントはモニタ保守ウィンドウ AIM をロードします。
- 収集フラグ (0x100) が付いた emphistory エントリが sysedge.cf ファイルに存在する場合、エージェントはパフォーマンス キューブ AIM をロードします。

CA SystemEDGE 5.0 より前のバージョンには、トップ プロセス AIM と呼ばれる AIM が含まれました。CA SystemEDGE 5.0 では、トップ プロセス AIM の機能はコア エージェントに分解されています。そのため、この AIM はロード可能なプラグインではなくなりました。

トップ プロセス AIM

以前のトップ プロセス AIM の機能は SystemEDGE コアに含まれています。sysedge.cf ファイルで「no_topprocs_table」ディレクティブをアクティブにするときは、トップ プロセス機能を無効にします。

モニタ保守ウィンドウ AIM

SystemEDGE モニタのアクティビティを時間ベースで制御するには、モニタ保守ウィンドウ AIM を使用します。この AIM は、セルフ モニタリング、プロセス モニタ、プロセス グループ モニタ、ログ ファイル モニタ、履歴、および Windows イベント モニタ (Windows の場合のみ) に適用されます。モニタ ウィンドウ AIM を使用すると、モニタ自体を停止しなくても大きな影響がない時間帯は、そのようなモニタからのトラップ (アラーム) の生成が停止されます。

モニタ保守ウィンドウ AIM の有効化

モニタ保守ウィンドウ AIM を起動するには、`sysedge.cf` ファイルで適切なエントリを指定する必要があります。SystemEDGE は、ウォーム スタートを実行し、AIM を自動的に起動します。

以下のエントリにより、モニタのモニタ (保守) 時間ウィンドウが指定されます。

構文

```
monwin monitor index timeOff timeOn status
```

monitor

モニタ テーブル (**selfmon**、**processmon**、**procgrouppmon**、**nteventmon**、**logfilemon**、**emphistory**) を指定します。

index

モニタのインデックスを指定します。

timeOff

モニタがオフになる (**notInService**) 時間を **24 時間形式 (hhmm)** で指定します。保守ウィンドウが開始します。

timeOn

モニタがオンになる (**active**) 時間を **24 時間形式 (hhmm)** で指定します。保守ウィンドウが終了します。

status

このエントリが **active** (1) か **notInService** (2) かを指定します。

例

```
monwin processmon 12 2200 2330 1
```

モニタ保守ウィンドウ AIM の制限事項

モニタ ウィンドウ AIM は日単位でのみ機能します。曜日、日付、または他の日単位以外のモニタ制御はサポートされません。

モニタ テーブルに実際には存在しない AIM 設定ファイルで指定されたモニタはすべて、存在する可能性が生じるまで無視されます。AIM が時間制御を実行するモニタはすべて、そのインデックスがモニタ テーブルから削除されている場合は無視されます。

AIM は、指定された正確な時間にモニタのアクティブ化および非アクティブ化を実行しようとします。ただし、負荷が高いシステムでは、AIM が正確な時間にサイクルを行わない場合があります。そのため、行のオフの時間とオンの時間の間隔は必ず 1 分より長くしてください。

AIM の設定ファイルでは、任意の 1 つのモニタが複数の行のターゲットとなる可能性があります。つまり、モニタは 1 日の間に複数回オフおよびオンになることがあります。この場合、オフとオンの時間が重複することによる論理的な問題をなくす責任はユーザにあります。

パフォーマンス キューブ AIM

パフォーマンス キューブ AIM は、履歴テーブル エントリを読み取り、キューブ 収集間隔（デフォルトでは 20 分）で平均メトリック計算を実行し、平均メトリック値をキューブ ファイル（パフォーマンス データ ファイル）に書き込みます。AIM は、既存の SystemEDGE 履歴テーブルを使用して、収集するメトリックを定義し、メトリック値の平均の計算を実行します。

パフォーマンス キューブ AIM の有効化

収集フラグ (0x100) が付いた `emphistory` エントリが `sysedge.cf` ファイルに 1 つ以上存在する場合、SystemEDGE はパフォーマンス キューブ AIM を暗黙的にロードします。AIM を使用するために、`sysedge.cf` に `sysedge_plugin` エントリは必要ありません。

構文

```
emphistory index interval oid number ['descr' [flags]]
```

`index`

使用するテーブルインデックスを指定します。

`interval`

30 秒の倍数で収集間隔を指定します。

`oid`

収集するオブジェクトの **OID** を指定します (ドット表記の完全な **OID** または `attrName.instOID` のいずれか)。

数値

収集するサンプル数を指定します。

`descr`

任意の説明を指定します。

`flags`

0x100 と指定された場合、パフォーマンス キューブ AIM が有効になります。

例

```
emphistory 11 60 loadAverage1Min.0 60 '1-min load average history' 0x100
```

第 18 章: コマンドライン ユーティリティ

この章では、SystemEDGE コマンドライン ユーティリティの使用方法について説明します。

このセクションには、以下のトピックが含まれています。

[SystemEDGE コマンドライン ユーティリティ \(P. 494\)](#)

[SystemEDGE のその他のコマンドライン ユーティリティ \(P. 495\)](#)

[SystemEDGE コマンドライン ユーティリティの設定 \(P. 495\)](#)

SystemEDGE コマンドライン ユーティリティ

管理対象外モードで SystemEDGE を実行する場合、SystemEDGE コマンドライン ユーティリティを使用して、CA SystemEDGE エージェントを操作するためのパラメータを設定できます。

SystemEDGE コマンドライン ユーティリティは、IPv4 および IPv6 環境の両方で SNMPv1、SNMPv2c、および SNMPv3 を使用して通信します。これらのユーティリティは、SystemEDGE のインストールディレクトリの bin サブディレクトリ内にあります。

ユーティリティによって渡される情報は、エージェントの設定ファイル（sysedge.cf および sysedgeV3.cf）内の情報と一致する必要があります。SystemEDGE 設定ファイルの詳細については、「エージェントの設定」の章を参照してください。

エージェントは、以下の SNMP コマンドライン ユーティリティをサポートします。

- diagsysedge
- edgemon
- edgewatch
- emphistory
- se_enc
- sendtrap
- snmpget
- snmpset
- sysvariable
- walktree
- edgetrapmon

SystemEDGE のその他のコマンドライン ユーティリティ

SystemEDGE は、SNMP ベースではないその他のコマンドライン ユーティリティを提供します。これらのユーティリティは、**SystemEDGE** モニタでアクションアイテムとして機能します。アクションは、モニタ テーブルエントリが **true** と表示されたときに実行されます。または、ステートフル モニタの場合は、しきい値違反の結果、関連する管理対象オブジェクトの状態が変化したときに実行されます。

エージェントは、以下の追加コマンドライン ユーティリティをサポートします。

- bounce (Windows)
- checkfile
- email
- getver (Windows)
- restartproc.exe (Windows)
- restartproc.sh (UNIX/Linux)
- restartsvc (Windows)

SystemEDGE コマンドライン ユーティリティの設定

SystemEDGE コマンドライン ユーティリティにアクセスすることにより、CA SystemEDGE エージェントを操作するためのパラメータを設定します。

次の手順に従ってください：

1. 管理者権限を使用して管理対象ノードにログインします。
2. コマンドプロンプト ウィンドウを開き、Install_Dir/SystemEDGE/bin ディレクトリに移動します。

これで、コマンドプロンプトで必要な SystemEDGE コマンドを入力できます。

注： SystemEDGE コマンドライン ユーティリティの詳細については、コンソールヘルプ オプション (-h) を使用してください。たとえば、edgemon については、コマンドプロンプトで以下のコマンドを入力します。

```
edgemon -h
```


第 19 章：状態管理の使用例

以下の例では、SystemEDGE の一般的な状態管理機能を使用します。

このセクションには、以下のトピックが含まれています。

[Windows タスク スケジューラのモニタ](#) (P. 497)

[隔離ファイルの存在のモニタ](#) (P. 499)

[ファイルシステムのモニタ](#) (P. 499)

[SNMP を介した拡張スクリプトの再設定](#) (P. 501)

[ログファイルのモニタと、電子メールによる通知の送信](#) (P. 505)

[モニタプロセス](#) (P. 506)

[モニタ対象オブジェクトの作成](#) (P. 507)

Windows タスク スケジューラのモニタ

この例では、Windows のタスク スケジューラで失敗したタスクまたは過剰なタスクをモニタします。sysedge.cf ファイルの一連のエントリと関連付けられた schedlgu.cmd スクリプトで構成されています。schedlgu.cmd スクリプトは *.job ファイルをカウントし、%SystemRoot%\Tasks ディレクトリの *.txt ファイルを分析し、%CASYSEDGE_DATA%\schedlgu.log ファイルに結果を書き込みます。SystemEDGE はスクリプトを起動して schedlgu.log ファイルをモニタし、現在のステータスを評価します。

schedlgu.cmd スクリプトを %CASYSEDGE%\bin ディレクトリにコピーするか、%CASYSEDGE%\bin\managedscripts ディレクトリにスクリプトを送信する設定展開パッケージにそのスクリプトを追加します。

sysedge.cf のエントリ

```
# Windows sample configuration for monitoring the Task Scheduler for failed or
excessive tasks
extension 55 integer readonly 'schedlgu.cmd %CASYSEDGE_DATA%\schedlgu.log'
watch logfile 55 0x0 '%CASYSEDGE_DATA%\schedlgu.log' 'The task completed with an exit
code of [()][0]*[^0].*[]]' 'Windows Task Scheduler Job Execution Errors' '' 10 warning
watch logfile 56 0x0 '%CASYSEDGE_DATA%\schedlgu.log' 'WARNING' 'Windows Task
Scheduler Warnings' '' 10 major
watch logfile 57 0x0 '%CASYSEDGE_DATA%\schedlgu.log' 'ERROR' 'Windows Task Scheduler
Errors' '' 10 critical
monitor oid extensionGroup.55.0 55 0x100 600 absolute > 15 'Number of Scheduled Tasks
> 15' '' 'SysHealth' 'Scheduled Tasks' 'Count' 'minor'
```

拡張スクリプト schedlgu.cmd

```
@echo off

REM SystemEDGE extension script for monitoring Windows Task Scheduler

setlocal

set SLOG=.%schedlgu.log
if "%1" NEQ "" set SLOG=%1

set SDIR=%SystemRoot%\Tasks

set SREC=
set DONE=N

REM Logical records in the Task Scheduler log file span multiple lines.
REM Convert into single line format (as %SLOG%) for monitoring via SystemEDGE.
if exist %SLOG% del %SLOG%
for /f "tokens=*" %i in ('type "%SDIR%\%schedlgu.txt"') do set
LINE=%i&call :parseline

REM Return number of configured tasks
set COUNT=0
for /f "tokens=*" %i in ('dir /b "%SDIR%\*.job"') do call :countline
echo %COUNT%

endlocal

goto :EOF

:countline
set /a COUNT+=1
goto :EOF

:parseline
REM Replace double quotes with pound sign
set LINE=%LINE:"=#%
REM Check if we have already processed the most recent record in the log
if "%DONE%" EQU "Y" goto :EOF
REM Check for start of next logical record
if "%LINE:~0,1%" EQU "#" goto :startrecord
if "%LINE%" EQU "[ ***** Most recent entry is above this line ***** ]" set
DONE=Y&goto :startrecord
REM Concatenate lines for current record
set SREC=%SREC% %LINE%
goto :EOF
```

```

:startrecord
if "%SREC%"==" " goto :initrecord
REM Restore double quotes
set SREC=%SREC:#"="
REM Write current record
@echo %SREC%>>"%SLOG%"

:initrecord
set SREC=%LINE%

```

隔離ファイルの存在のモニタ

この例では、%ProgramFiles%\CA\TrustITM\Move ディレクトリ内の eTrust Antivirus によって隔離されたファイルの存在を 10 分おきにモニタします。この例は、sysedge.cf ファイルの 2 つのエントリで構成されており、logfile パラメータを使用してディレクトリのエントリをモニタします。0x1808 フラグを付けると、ディレクトリが再帰的にモニタされ、トラップがまったくログ記録されなくなります。

```

# Windows sample configuration for monitoring the existence of any files quarantined
by eTrust Antivirus
watch logfile 45 0x1808 '%ProgramFiles%\CA\TrustITM\Move' '' 'Monitor eTrust
Antivirus Quarantine directory' '' 10 warning
monitor logMonitorEntry '%ProgramFiles%\CA\TrustITM\Move' logMonitorLogFileCount
45 0x100 600 absolute > 0 'eTrust Antivirus Quarantine directory not empty' '' '' ''
'' 'major'

```

ファイル システムのモニタ

この例では、Windows 上のファイル システムをモニタします。この例は、sysedge.cf ファイルの以下の 3 つのグループのエントリで構成されています。

- しきい値によって重大度が変わるオブジェクト状態モデルの使用。
- 同じクラス、インスタンス、属性、および重大度と組み合わせたフラグ 0x800 の使用。フラグ 0x800 により、重大度が同じすべてのモニタの集計された状態が指定され、AND 関係が適用されます。
- aggregate_level 制御の使用。

注: デフォルトでオブジェクト インスタンスに追加されるプレフィックス //./ (localhost を表します) は、この例では省略されています。

```
# Derives a state for file system C: (System).
# Creates an aggregate 'FSys' 'System' 'PercentUsed', which can be in state ok, minor,
major, or fatal.

monitor devTableEntry 'C:' devCapacity 51 0x0 120 absolute > 90 'Description' '' 'FSys'
'System' 'PercentUsed' minor
monitor devTableEntry 'C:' devCapacity 52 0x0 120 absolute > 95 'Description' '' 'FSys'
'System' 'PercentUsed' major
monitor devTableEntry 'C:' devCapacity 53 0x0 120 absolute > 99 'Description' '' 'FSys'
'System' 'PercentUsed' fatal

# Derives warning state if three conditions are met simultaneously, that is, only if
all three Backup file systems E:, F:, and G: are above 95%.
# Creates an aggregate 'FSys' 'Backup' 'PercentUsed', which can be in state ok or
warning.

monitor devTableEntry 'E:' devCapacity 54 0x800 120 absolute > 95 'Description' ''
'FSys' 'Backup' 'PercentUsed' warning
monitor devTableEntry 'F:' devCapacity 55 0x800 120 absolute > 95 'Description' ''
'FSys' 'Backup' 'PercentUsed' warning
monitor devTableEntry 'G:' devCapacity 56 0x800 120 absolute > 95 'Description' ''
'FSys' 'Backup' 'PercentUsed' warning

# Derives worst state of all file systems.
# Creates a super aggregate 'FSys' '*' 'PercentUsed', which can be in state ok, warning,
minor, major, or fatal.

aggregate_level 0x0002
```

SNMP を介した拡張スクリプトの再設定

これらのスクリプトを使用すると、SNMP を使用してリモートで再設定できる拡張スクリプトを使用して、任意のシステムに ping できます。そのため、sysedge.cf や他のカスタム スクリプトを編集する必要がありません。任意の MIB ブラウザまたは SystemEDGE に用意された event snmpset ユーティリティから SNMP SET を使用することで、監視するシステムの名前を設定または変更できます。拡張変数を読み取ると、ping 情報を含む文字列が取得されます。

SNMP を介して拡張スクリプトを再設定する方法

1. sysedge.cf 設定ファイルに以下の行を追加して、SystemEDGE を再起動します。

```
extension 31 octetstring readwrite '/opt/CA/test/ping.sh' (UNIX, Linux)
extension 31 octetstring readwrite 'c:¥test¥ping.bat' (Windows)
```

2. 拡張ディレクティブで指定したディレクトリにスクリプト (ping.sh、ping.bat) をコピーします。

3. ping のターゲットを設定に、読み取り要求を入力します。

```
C:¥SystemEDGE¥bin>.¥snmpget -o 1.3.6.1.4.1.546.14.31.0
1.3.6.1.4.1.546.14.31.0 Ping target must be set first
```

エラー メッセージが表示されます。

4. my-sles3 を、ping するシステムに設定します。

```
C:¥SystemEDGE¥bin>.¥snmpset -c private -o 1.3.6.1.4.1.546.14.31.0 -s my-sles3
1.3.6.1.4.1.546.14.31.0 my-sles3
```

5. 変数を読み取ることでシステムに ping します。

```
C:¥SystemEDGE¥bin>.¥snmpget -o 1.3.6.1.4.1.546.14.31.0
1.3.6.1.4.1.546.14.31.0 my-sles3 is alive
```

スクリプトは、拡張にインデックス 31 があることを確認します。ただし、この確認は技術的に必要でないため削除できます。その結果、拡張スクリプトは任意のインデックスに割り当てることができます。

スクリプト ping.bat および ping.sh により、設定情報が一時ファイルに格納されます。

UNIX および Linux の ping スクリプト

```
#!/bin/sh
#
# ping.sh example extension script
#
# A simple example script for the SystemEDGE agent extension group.
# This script implements a "Ping" SNMP MIB variable whereby
# Sets to this variable indicate what system should be pinged
# and Gets to this variable cause the Ping to occur and the
# output to be returned as an OctetString.
#
# For reference, the following are passed as parameters by the SystemEDGE
# Agent. They are not used in this script.
#
# The Extension Group Leaf Number
LeafNumber=$1
# The SNMP Operation: GET, GETNEXT, or SET
Operation=$2
# The value if a SET Operation
SetValue=$3

# determine which ping program to use
OSNAME=`/bin/uname -r`
OSTYPE=`/bin/uname -s`

if [ "$LeafNumber" = "31" ]; then

    # setup who we should be pinging
    if [ "$Operation" = "SET" ]; then
        echo $SetValue > /tmp/sysedge.ping.target
        echo $SetValue
    fi

    if [ "$Operation" = "GETNEXT" ]; then
        Operation="GET"
    fi

    if [ "$Operation" = "GET" ]; then
        if [ ! -f /tmp/sysedge.ping.target ]; then
            echo "Ping target must be Set first"
            exit 0
        fi
    fi
fi
```

```

case $OSTYPE in
    Linux)
        /bin/ping -c 1 -q -n `/bin/cat /tmp/sysedge.ping.target`
        ;;

    HP-UX)
        /etc/ping `/bin/cat /tmp/sysedge.ping.target` -n 1
        ;;

    AIX)
        /etc/ping -c 1 `/bin/cat /tmp/sysedge.ping.target`
        ;;

    SunOS)
        /usr/sbin/ping `/bin/cat /tmp/sysedge.ping.target` 2
        ;;

    *)
        case $OSNAME in
            6.*)
                /usr/etc/ping -c 1 -q -n `/bin/cat /tmp/sysedge.ping.target`

                # IRIX PING は非常に面倒です。マシンがアライブかどうかを示している出力を
                # 1 行取得できるのですが、
                # 取得できているようには見えないからです。
                # 出力を捕捉し、最後の行のみ echo するのも
                # 1 つの方法です。
                ;;

            5.*)
                /usr/sbin/ping `/bin/cat /tmp/sysedge.ping.target` 2
                ;;

            4.*)
                /usr/etc/ping `/bin/cat /tmp/sysedge.ping.target` 2
                ;;

            *)
                echo 「システム用の PING コマンドを判断できません」 $OSNAME
                exit 0
                ;;
        esac
    esac
fi
fi

```

Windows 用 PING スクリプト

```
@ECHO OFF
@rem ping.bat のサンプル拡張スクリプト
@rem
@rem SystemEDGE エージェント拡張グループ用の簡単なスクリプト例。
@rem このスクリプトは、SNMP MIB 変数、「Ping」を実装します。実装することにより、
@rem この変数に対する Set は、PING する必要があるシステムを示し、
@rem この変数に対する Get は、PING を発生させ、
@rem 出力を OctetString として返します。
@rem
@rem 参考までに、SystemEDGE エージェントによって、以下がパラメータとして
@rem 渡されます。
@rem

@rem 拡張グループのリーフ番号
set LeafNumber=%1%

@rem SNMP の操作： GET、GETNEXT、または SET
set Operation=%2%

@rem SET 操作の場合の値
set SetValue=%3%

@rem 使用する必要がある PING コマンド
set PING=%SystemRoot%\system32\ping.exe

@rem 後で使えるよう、「環境」内のターゲットを
@rem 変数 SYSEdge_PING_TARGET を使用して保存します。

if not %LeafNumber% == 31 goto fi

@rem Set の場合は、値があることを確認し、
@rem 保存してから、値を STDOUT に echo します。
@rem これは、拡張コードをわかりやすくするためです。
if %Operation% == SET if !%SetValue%==! goto error
if %Operation% == SET @echo set SYSEdge_PING_TARGET=%SetValue% > %TMP%\pingtgt.bat
if %Operation% == SET echo %SetValue%
if %Operation% == SET goto fi
```



```
@rem スクリプトを実行して、ターゲット名を回復します。
call %TMP%\pingtgt.bat
if !%SYSEdge_PING_TARGET% ==! goto error

@rem %SYSEdge_PING_TARGET% の PING を echo
%PING% -n 1 -w 250 %SYSEdge_PING_TARGET% > nul

if ERRORLEVEL 1 echo %SYSEdge_PING_TARGET% not alive
if not ERRORLEVEL 1 echo %SYSEdge_PING_TARGET% is alive

:fi
goto exit

:error
echo PING ターゲットを最初に設定する必要があります。

:exit
rem command.com には欠陥が多いので、クリーンアップを実行
set PING=
set Operation=
set LeafNumber=
set SetValue=
```

ログ ファイルのモニタと、電子メールによる通知の送信

この例では、Windows 修正プログラムのログ ファイルをモニタして、「fatal」という文字列がないかどうかを確認します。「fatal」という文字列がある場合、SystemEDGE は電子メールによって通知を送信します。この例は、sysedge.cf ファイルのエントリと、関連する Send_Notification_Mail.cmd スクリプトから構成されています。

注: #no_actions ディレクティブはアクティブにしないでください。先頭にコメント (#) 文字を付加してください。

```
watch logfile 44 0x8 '%windir%\KB*.log' 'fatal' 'check KB for fatal occurrence in MS
hotfix log' "%CASYSEdge%\My_Scripts\Send_Notification_Mail.cmd" 1 warning
```

Send_Notification_Mail.cmd スクリプト

```
rem *** 関連する引数を取得 ***
set regexpr=%3
set lasttime=%5
set description=%6

rem *** 引数 %10 を %9 として取得するためにシフト ***
shift
set current_file=%9

rem *** 電子メールを呼び出す ***
"c:\Program Files\CA\SystemEDGE\bin\email.exe" -r smtp.company.com sysedge@%
COMPUTERNAME% you@company.com 'KB log fatal found' 'File monitor %description%
activated. %regexpr% pattern found in %current_file% at time %lasttime%'
```

モニタプロセス

引数を含めてプロセスをモニタ（フラグ値に注目）し、管理対象オブジェクトを作成します（Class=notepad、Instance=bad、Attribute=alive、state=severity）。

```
watch process procAlive 'notepad.*bad.*' 122 0x800 30 absolute nop 0 'Description:
procAlive(notepad)' '' 'notepad' 'bad' 'alive' 'minor'
```

一群のプロセスのプロパティをモニタするために **procgrou**p を作成します。そうすると、**proggrou**p の属性（RSS= 実メモリ常駐セットなど）を、**Monitor** ディレクティブを使用して、他の **OID** のようにモニタできます。

```
watch procgroup 'svchost' 222 0x0 30 'Descr: procgroup(svchost)' ''
monitor processGroupMonEntry svchost pgmonRSS 20 0x0 60 absolute > 15000 'Descr:
procgroup svchost' '' ProcHealth svchost pgmonRSS critical
```

モニタ対象オブジェクトの作成

特定の修正プログラムがインストールされているかどうかを確認し、インストールされている場合は、モニタ対象オブジェクトを作成します。

```
monitor hrSWInstalledEntry 'Microsoft Hotfix Patch KB958644' hrSWInstalledIndex 41  
0x100 30 absolute = 0 'Descr: Hot' '' 'SysHealth' 'SWInstalled' 'Microsoft Hotfix Patch  
KB958644' 'ok'
```

PerfObject、「UDpv4」のパフォーマンス カウンタ「Datagrams Sent/sec」の値を保持する MIB 属性を作成します。

注: 利用可能なパフォーマンス オブジェクトとカウンタについては、Microsoft のコマンド `perfmon` で確認してください。

```
ntregperf 2 counter performance 'UDpv4' 'Datagrams Sent/sec' ''
```


第 20 章: トラブルシューティングと使用方法に関する推奨事項

この章では、SystemEDGE エージェントの使用に役立つヒントについて説明します。

Windows では、Windows コントロール パネルから以下のすべての作業を行うことができます。

- エージェントの基本の表示：システム情報と SNMP コミュニティ
- エージェント設定ファイルの表示と変更
- エージェント診断ファイルの表示
- ホストの起動と停止

このセクションには、以下のトピックが含まれています。

[diagsysedge.exe の使用](#) (P. 509)

[よくある問題と質問](#) (P. 512)

diagsysedge.exe の使用

エージェントが実行されていることを確認し、トラブルシューティングに使用できるエージェントに関する情報を取得するには、diagsysedge.exe プログラムを使用します。

このセクションの例では、エージェントが SNMPv1 または SNMPv2c を使用するよう設定されており、パブリックの読み取りコミュニティがあり、ポート 161 または 1691 で実行されていることを想定しています。

注: diagsysedge.exe によりサポートされるすべてのオプションの詳細については、「コマンドライン ユーティリティ」の章を参照してください。

diagsysedge によるエージェントが実行されているかどうかの確認

-B オプションを付けて **diagsysedge.exe** を実行すると、エージェントが実行されているかどうかを確認できます。

エージェントが実行されているかどうかを確認する方法

1. コマンドプロンプト ウィンドウを開き、**SystemEDGE** の **bin** ディレクトリに移動します。
2. 以下の情報を入力します。

UNIX :

```
./diagsysedge -B
```

注: **diagsysedge** の代わりに、UNIX でシンボリック リンク **diagsysedge.exe** を使用してユーティリティを実行することもできます。

Windows :

```
diagsysedge.exe -B
```

このコマンドにより、次のような出力が生成されます。

```
#!/diagsysedge.exe -B
diagsysedge: (V1.03 - LINUX)
egyptian
2.6.9-11.ELsmp, Red Hat Enterprise Linux AS release 4 (Nahant Update 1)#1 SMP Fri
May 20 18:26:27 EDT 2005
```

エージェントが実行されていない場合、**SNMP** 操作が失敗したというエラーメッセージを受け取ります。ポート番号を確認し、エージェントが **161** 以外のポートで起動している場合は、コマンドラインで **-p port** を指定してやり直すか、「エージェントの起動」の章の手順を使用してエージェントの起動をもう一度試みることができます。

トラブルシューティング用のレポートの取得

SystemEDGE で問題が生じた場合、diagsysedge.exe を実行して、トラブルシューティングに使用できるレポートを生成するようテクニカル サポートが願います。

テクニカル サポートに送信可能なレポートを生成する方法

1. コマンドプロンプトを開き、sysedge/bin ディレクトリに移動します。
2. 以下の情報を入力します。

UNIX :

```
./diagsysedge
```

注: diagsysedge の代わりに、UNIX でシンボリック リンク diagsysedge.exe を使用してユーティリティを実行することもできます。

Windows :

Windows コントロール パネルを使用して、diagsysedge 出力を作成します。

エージェントが実行されていない場合、SNMP 操作が失敗したというエラーメッセージを受け取ります。使用しているオペレーティング システムに応じたこの章の手順を使用して、エージェントの起動をもう一度試みます。

AIM プラグインの検証

-Q オプションを付けて diagsysedge.exe を実行すると、AIM が実行されているかどうかを確認できます。

AIM が実行されているかどうかを確認する方法

1. コマンドプロンプト ウィンドウを開き、SystemEDGE の bin ディレクトリに移動します。

2. 以下の情報を入力します。

UNIX/Linux :

```
./diagsysedge -Q
```

注:diagsysedge の代わりに、UNIX でシンボリック リンク diagsysedge.exe を使用してユーティリティを実行することもできます。

Windows :

```
diagsysedge.exe -Q
```

このコマンドにより、次のような出力が生成されます。

```
C:\Program Files\CA\SystemEDGE\bin>diagsysedge -Q
--- diagsysedge plugin query result ---
monwin : RESULT: ok BRANCH: 1.3.6.1.4.1.546.16.20 DESCR: CA SystemEDGE AIM for
Monitor Timing Windows, VERSION:5.0.0 BUILD:10092 REQUIRES:5.0.0
perfcube : RESULT: ok BRANCH: 1.3.6.1.4.1.546.16.21 DESCR: CA SystemEDGE AIM for
Performance Cubes, VERSION:5.0.0 BUILD:10092 REQUIRES:5.0.0
cavmvcaim : RESULT: ok BRANCH: 1.3.6.1.4.1.546.16.52 DESCR: CA VMware VC Server
AIM 2.0 PL0
rmonwbem : RESULT: ok BRANCH: 1.3.6.1.4.1.546.16.22 DESCR: SysEDGE RMONWBEM AIM,
VERSION:1.0.0 BUILD:10102 REQUIRES:5.0.0
```

```
C:\Program Files\CA\SystemEDGE\bin>
```

エージェントが実行されていない場合、SNMP 操作が失敗したというエラーメッセージを受け取ります。ポート番号を確認し、エージェントが 161 以外のポートで起動している場合は、コマンドラインで **-p port** を指定してやり直すか、「エージェントの起動」の章の手順を使用してエージェントの起動をもう一度試みることができます。

よくある問題と質問

このセクションでは、SystemEDGE をインストールして使用するときに生じる可能性がある問題について説明します。さらに、これらの問題を解決するための手順についても説明します。このセクションは、問題の目に見える現象と、原因を隔離するために実行可能な検証ステップごとに整理されています。

トラブルシューティング用の診断ログ情報

SystemEDGE で問題が生じた場合、sysedge.cf ファイルを編集することで診断ログ記録設定を調整する必要があります。sysedge.cf ファイルを編集する手順については、「診断ロギング メカニズムの設定」の章で説明されています。

通常、ログ記録される情報の量は増えるため、ログ ファイルの最大サイズとログ ファイルの数を調整する必要があります。

例：

```
sysedge_logfile sysedge.log 20000 2
sysedge_loglevel debug
```

SystemEDGE には、それぞれ最大サイズが 20000 KB の 2 つの追加ログ ファイルのセットを使用するように指示が与えられます。

エージェントが SNMP リクエストに回答しない

エージェントが SNMP リクエストに回答していない場合 この問題を解決するには、以下の手順に従って原因を特定する必要があります。

- [エージェントが実行されていることを確認します。](#) (P. 514)
- [エージェントがシステムの初期化時に起動することを確認します。](#) (P. 514)
- [エージェントがクエリに回答することを確認します。](#) (P. 515)
- エージェント バイナリおよび設定ファイルがインストールされていることを確認します。
- エージェントが他のエージェントと正しく共存していることを確認します。
- [エージェント設定を確認します。](#) (P. 516)
- [管理ソフトウェア設定を確認します。](#) (P. 516)

エージェントが実行中であることの確認

プラットフォームに応じて、エージェントが以下のいずれかの方法で実行されていることを確認できます。

UNIX でエージェントが実行されていることを確認する方法

1. 「netstat -a」と入力し、UDP/161、UDP/1691、またはSNMPを探します。
2. 「ps -aux」または「ps -aef」と入力し、sysedgeを探します。
3. walktree または snmpget を実行します。
4. sysedge.log ファイルで、エージェント エラー メッセージがないか調べます。

Windows でエージェントが実行されていることを確認する方法

1. [スタート]、[コントロールパネル]、[CA SystemEDGE] の順に選択し、ステータスを表示します。
2. Windows タスク マネージャを開き、[プロセス] タブを選択します。
sysedge.exe という名前のプロセスを探します。このプロセスがリストに表示される場合、CA SystemEDGE は実行されています。
3. [スタート]、[コントロールパネル]、[管理ツール]、[サービス] の順に選択し、[サービス] ダイアログボックスで「CA SystemEDGE」を探します。
4. walktree または snmpget を実行します。
5. sysedge.log ファイルで、エージェント エラー メッセージがないか調べます。

システムの初期化時におけるエージェントの起動の確認

エージェントがシステムの初期化時に起動することを確認するには、プラットフォームに応じて以下のいずれかを実行します。

- UNIX 起動スクリプトを確認します。
- Windows SystemEDGE サービスで自動的起動が設定されていることを確認します。

クエリに対するエージェントの応答の確認

sysvariable ユーティリティを使用してシステムをクエリし、エージェントがクエリに応答することを確認します。エージェントが **SNMPv1** または **SNMPv2c** 通信を受け入れるように設定されている場合、エージェントのポート番号（デフォルトのポート **161** を使用していない場合）およびシステムのコミュニティ文字列を指定する必要があります。指定しない場合、必要な **SNMPv3** 通信（セキュリティレベルに応じて、**Authpasswd**、**Authprotocol**、**Privpasswd**、**Privprotocol**）をすべて指定する必要があります。

エージェントがポート **1691** で実行されており、コミュニティ文字列がパブリックになっている **UNIX** システムのオペレーティングシステム リリース番号を表示するには、**SNMPv1** の場合は以下の内容を入力します。

```
#./sysvariable -h 127.0.0.1 -p 1691 -c public -v 1 -o systype
```

エージェントがポート **1691** で実行されており、コミュニティ文字列がパブリックになっている **UNIX** システムのオペレーティングシステム リリース番号を表示するには、**SNMPv2c** の場合は以下の内容を入力します。

```
#./sysvariable -h 127.0.0.1 -p 1691 -c public -v 2c -o systype
```

エージェントがポート **1691** で実行されており、**SecLevel** が **AuthPriv(3)**（**USM** ユーザ名が **secuser**、認証パスワードが **authpasswd**、認証プロトコルが **MD5**、プライバシーパスワードおよび **DES** が **privpasswd**）に設定されている **UNIX** システムのオペレーティングシステム リリース番号を表示するには、**SNMPv3** の場合は以下の内容を入力します。

```
#./sysvariable -h 127.0.0.1 -p 1691 -v 3 -u secuser -s 3 -a authpasswd -A MD5 -x privpasswd -X DES -o systype
```

エージェントがポート **1691** で実行されており、コミュニティ文字列がパブリックになっている **Windows** システムでは、**SNMPv1** の場合は以下の内容を入力します。

```
C:\sysedge\bin>sysvariable.exe -h 127.0.0.1 -p 1691 -c public -v 1 -o systype
```

エージェントがポート **1691** で実行されており、コミュニティ文字列がパブリックになっている **Windows** システムでは、**SNMPv2c** の場合は以下の内容を入力します。

```
C:\sysedge\bin>sysvariable.exe -h 127.0.0.1 -p 1691 -c public -v 2c -o systype
```

エージェントがポート **1691** で実行されており、**SecLevel** が **AuthPriv(3)**（**USM** ユーザ名が **secuser**、認証パスワードが **authpasswd**、認証プロトコルが **MD5**、プライバシーパスワードおよび **DES** が **privpasswd**）に設定されている **Windows** システムでは、**SNMPv3** の場合は以下の内容を入力します。

```
C:\sysedge\bin>sysvariable.exe -h 127.0.0.1 -p 1691 -v 3 -u secuser -s 3 -a authpasswd -A MD5 -x privpasswd -X DES -o systype
```

エージェント設定の確認

エージェントが正しく設定されていることを確認するには、アクセス制御リストを確認します。SNMPv2 と SNMPv2c の場合は、sysedge.cf でコミュニティを確認します。SNMPv3 の場合、sysedgeV3.cf で USM セキュリティ設定を確認します。

以下のように、プラットフォームごとにこれらの検証を実行します。

- UNIX システムでは、コミュニティ文字列には SYSEGE_DATADIR/sysedge.cf、USM セキュリティ設定には SYSEGE_DATADIR/sysedgeV3.cf を実行します。
- Windows システムでは、[CA SystemEDGE コントロール パネル] を介して SNMP v1 および SNMPv2c コミュニティを確認します。
SYSEGE_DATADIR¥sysedgeV3.cf で USM セキュリティ情報を確認します。

注:sysedgeV3.cf は暗号化されている場合があります。

詳細については、「コマンドライン ユーティリティ」の「se_enc ユーティリティ --SNMPV3 設定ファイルの暗号化」を参照してください。

管理システム設定の確認

管理システム ソフトウェアが正しく設定されていることを確認するには、SystemEDGE が代替 UDP ポート（たとえば、UDP/1691 など）で実行されている場合に、適切なシステムおよびポート番号をクエリしていることを確認します。

管理システムが SNMP トラップ メッセージを受信しない

管理システムが SystemEDGE から SNMP トラップを受信しない場合、エージェントがトラップ PDU を送信しており、それらを適切なアドレスに送信していることを確認する必要があります。すでにそれらの条件を確認している場合、問題は管理システムの設定にあると思われます。以下の検証を実行して、その問題の根本原因を隔離します。

- [エージェントが実行されていることを確認します。](#) (P. 514)
- [エージェントがトラップ メッセージを送信するように設定されていることを確認します。](#) (P. 517)
- [エージェントがトラップを送信することを確認します。](#) (P. 517)
- [管理システムのトラップ設定を確認します。](#) (P. 517)

エージェントのトラップ設定の確認

適切なアドレスにトラップメッセージを送信するようにエージェントが設定されていることを確認するには、以下のいずれかを実行します。

- SNMPv1 の場合、`sysedge.cf` ファイルでトラップ コミュニティを確認します。
- SNMPv3 の場合、`sysedgeV3.cf` ファイルで USM セキュリティ設定を確認します。
- UNIX システムでは、`sysedge.cf` ファイルでトラップ コミュニティを確認します。たとえば、各トラップ コミュニティ ステートメントに指定されている IP アドレスが 1 だけであることを確認します。トラップ コミュニティ設定の詳細については、「エージェントの設定」の章を参照してください。
- Windows システムでは、コントロールパネルアプリケーションを介してトラップ設定（SNMPv1 のみ）を確認します。詳細については、「エージェントの設定」の章の「Configure Trap Communities」を参照してください。

エージェントがトラップを送信することの確認

エージェントがトラップを送信することを確認するには、以下の手順に従います。

- UNIX システムでは、`CASYSEGE_DATADIR/sysedge.log` を調べます。
- Windows システムでは、`CASYSEGE_DATADIR¥sysedge.log` を調べます。

管理システムのトラップ設定の確認

管理システムの設定が適切であることを確認するには、`empire.asn1` ファイルが管理ステーションにインポートされていることを確認します（その結果、管理システムは SystemEDGE エージェントによって生成された SNMP トラップ PDU の形式を識別できます）。管理システム ソフトウェアのトラブルシューティングの詳細については、管理システムのベンダーに問い合わせてください。

バインド失敗:アドレスがすでに使用中

アドレスが使用中のためバインドが失敗した場合、エージェントには最初の起動時に次のようなメッセージが表示されます。

```
sysedge: bind call failed: Address already in use  
sysedge: another agent is probably running on port X
```

このメッセージは、別の **SNMP** エージェントがすでに実行されていて、デフォルトの **SNMP** エージェントポートである **UDP/161** にバインドされているときによく発生します。

他の **SNMP** エージェントと共に **SystemEDGE** エージェントを使用する方法の詳細については、付録「他の **SNMP** エージェントとの共存」を参照してください。代替 **UDP** ポート上でエージェントを使用する方法の詳細については、「エージェントの起動」の章を参照してください。

CA SystemEDGE: xtrapmon.exe ユーティリティは edgetrapmon.exe に置き換え

Windows、Linux、および UNIX では、**xtrapmon** が **edgetrapmon** ユーティリティに置き換えられました。**xtrapmon** ユーティリティは使用できなくなりました。**SystemEDGE** リリース ノートも参照してください。

付録 A: FIPS 140-2 の暗号化

この付録では、SystemEDGE の FIPS モードをインストールして有効にする方法について説明します。

このセクションには、以下のトピックが含まれています。

[FIPS 140-2 モード \(P. 519\)](#)

[FIPS ライブラリのインストール \(P. 519\)](#)

[プラットフォーム サポート \(P. 520\)](#)

[サポートされる暗号化プロトコル \(P. 520\)](#)

[サポートされる認証プロトコル \(P. 521\)](#)

[FIPS 140-2 モードの設定 \(P. 521\)](#)

[FIPS モードの考慮事項 \(P. 522\)](#)

[キー保護 \(P. 522\)](#)

FIPS 140-2 モード

米国連邦政府の規制では、製品に暗号化が含まれる場合、米国連邦政府への新しいソフトウェア製品販売すべてで連邦情報処理標準（FIPS 140-2）によって検証された暗号化アルゴリズムを使用することが求められます。

FIPS 140-2 標準の規則に従って認定された暗号化ライブラリのバージョンを使用して、SystemEDGE を FIPS モードで実行できます。

FIPS ライブラリのインストール

FIPS モードで実行するには、SystemEDGE コア パッケージに加えて SystemEDGE Advanced Encryption パッケージをインストールする必要があります。SystemEDGE Advanced Encryption パッケージは、コア エージェント パッケージより高度な暗号化を提供するライブラリと共に FIPS ライブラリをインストールします。

SystemEDGE Advanced Encryption パッケージによって、以下の FIPS 認定ファイルがインストールされます。

Windows

- cryptocme2.dll (FIPS ライブラリ)
- cryptocme2.sig (FIPS ライブラリ署名ファイル)

UNIX

- libcryptocme2.so (HP-UX PA-RISC を除く UNIX の FIPS ライブラリ)
- libcryptocme2.sl (HP-UX PA-RISC の FIPS ライブラリ)
- libcryptocme2.sig (FIPS ライブラリ署名ファイル)

SystemEDGE Advanced Encryption をインストールする方法の詳細については、付録「SystemEDGE Advanced Encryption」を参照してください。

プラットフォーム サポート

RSA B-safe Crypto FIPS 準拠のライブラリは、現在サポートされる以下のエージェントプラットフォームでのみ利用可能です。

- Windows 32 ビット
- Windows 64 ビット (Itanium 以外)
- Red Hat 32 および 64 ビット (Itanium 以外)
- SuSE Linux 32 および 64 ビット (Itanium 以外)
- Solaris SPARC
- Solaris 32 および 64 ビット (Intel 以外)

注: Solaris Intel アーキテクチャ プラットフォームでは、FIPS 140-2 がサポートされません。

- HP-UX IA-64
- HP-UX 11.31 PA-Risc 64 ビット

注: FIPS 140-2 は以前のバージョンの HP-UX PA-Risc ではサポートされません。

- AIX 32 および 64 ビット

サポートされる暗号化プロトコル

FIPS モードの動作では、以下の暗号化プロトコルが SystemEDGE によりサポートされます。以下にリストされていないプロトコルはサポートされません。

- トリプル データ暗号化標準 (3DES)
- 128 または 256 ビットの暗号化キーを使用した高度暗号化基準 (AES)

サポートされる認証プロトコル

FIPS のみのモードの動作では、以下の認証プロトコルが SystemEDGE によりサポートされます。

- Secure Hash Algorithm-1 (SHA)

FIPS 140-2 モードの設定

`sysedge.cf` ファイルで `sysedge_fips_mode` パラメータを使用して、エージェントが暗号化をどのように処理するかを設定できます。

`sysedge_fips_mode` パラメータのオプションは以下のとおりです。

0(ゼロ)

非 FIPS モードを示します。エージェントは、CA eTrust 公開鍵インフラストラクチャ ライブラリを有効にします。この方法が失敗した場合、認定されていない FIPS コードに基づく組み込みの基本暗号化および認証プロトコルを使用します。

1

FIPS 共存モードを示します。エージェントは、FIPS 認定ライブラリを使用して、FIPS 認定暗号化および認証プロトコルを使用しようとします。ただし、FIPS 認定ライブラリが見つからない場合や、エージェントがこれらのライブラリのロードに失敗した場合、エージェントの機能は FIPS に準拠しない CA eTrust 公開鍵インフラストラクチャ ライブラリの使用に戻ります。`sysedge_fips_mode` が設定されていない場合、これがデフォルトのモードです。

2

FIPS のみのモードを示します。FIPS 認定の暗号化と認証コードおよびプロトコルのみサポートされ、認定されていない FIPS コードおよびプロトコルはすべて許可されません。FIPS ライブラリが失敗する場合、暗号化は実行されません。

例

FIPS のみのモードでエージェントを実行するには、`sysedge.cf` ファイルに以下の行を追加します。

```
sysedge_fips_mode 2
```

FIPS モードの考慮事項

FIPS モードを使用するときは、以下の考慮事項に注意してください。

- FIPS モードがサポートされないプラットフォームで、エージェントが FIPS モード (`sysedge_fips_mode 1` または `sysedge_fips_mode 2`) で有効な場合、その動作は FIPS モードがサポートされるプラットフォームと同じです。

たとえば、FIPS のみのモード (`sysedge_fips_mode 2`) を有効にした場合と、FIPS モードがサポートされないプラットフォームで認証プロトコル MD5 が使用されている場合は、MD5 を使用したアクセスが失敗します (FIPS のみのモードがプラットフォームでサポートされていた場合と同じ動作です)。

キー保護

SystemEDGE では、設定ファイル `sysedgeV3.cf` を使用して SNMPv3 ユーザ設定を行うことができます。このファイルは、キーとパスワードを保護するために暗号化できます。詳細については、「エージェントの設定」の章を参照してください。

付録 B: SystemEDGE Advanced Encryption

この章では、追加の暗号化と SystemEDGE の FIPS 140-2 モードでの実行を可能にするため、SystemEDGE Advanced Encryption パッケージをインストールする方法について説明します。

このセクションには、以下のトピックが含まれています。

[SystemEDGE Advanced Encryption](#) (P. 523)

[サポートされるプラットフォーム](#) (P. 523)

[サポートされる暗号化プロトコル](#) (P. 523)

[サポートされる認証プロトコル](#) (P. 524)

[FIPS の互換性](#) (P. 524)

[インストールの前提条件](#) (P. 524)

[SystemEDGE Advanced Encryption のインストール](#) (P. 524)

[インストールされるファイル](#) (P. 527)

SystemEDGE Advanced Encryption

SystemEDGE Advanced Encryption には、SystemEDGE コア パッケージに用意されている暗号化に対する追加の暗号化が用意されています。このパッケージには、追加の暗号化を可能にし、FIPS モードで実行するための RSA BSAFE Crypto ライブラリが含まれる CA eTrust 公開鍵インフラストラクチャ (ETPKI) ライブラリが含まれています。

サポートされるプラットフォーム

SystemEDGE Advanced Encryption は、Solaris 8 および 9 の 32 ビット オペレーティング システムを除いて、SystemEDGE でサポートされるすべてのプラットフォームでサポートされ、インストール可能です。

サポートされる暗号化プロトコル

SystemEDGE Advanced Encryption パッケージがインストールされている場合、以下の暗号化プロトコルが SystemEDGE によってサポートされます。

- データ暗号化標準 (DES)
- トリプル データ暗号化標準 (3DES)
- 128 および 256 ビットの暗号化キーを使用した高度暗号化基準 (AES)

サポートされる認証プロトコル

SystemEDGE Advanced Encryption パッケージがインストールされている場合、以下の暗号化プロトコルが SystemEDGE によってサポートされます。

- Message-Digest algorithm 5 (MD5)
- Secure Hash Algorithm-1 (SHA)

FIPS の互換性

SystemEDGE Advanced Encryption パッケージは、RSA B-safe Crypto FIPS 準拠のライブラリを自動的にインストールします。FIPS モードによりサポートされるプラットフォームと具体的な考慮事項の詳細については、付録「FIPS 140-2 の暗号化」を参照してください。

インストールの前提条件

SystemEDGE のインストールは、SystemEDGE Advanced Encryption のインストールの前提条件です。

SystemEDGE Advanced Encryption のインストール

以下のセクションでは、Windows と UNIX で SystemEDGE Advanced Encryption をインストールおよびアンインストールする方法について説明します。

Windows での SystemEDGE Advanced Encryption のインストール

SystemEDGE Advanced Encryption パッケージは、サポートされるすべての Windows ハードウェア アーキテクチャで利用できます。インストールを実行すると、システム アーキテクチャに適したバージョンが自動的にインストールされます。

Windows で SystemEDGE Advanced Encryption をインストールする方法

1. Windows システムに管理者としてログオンし、
DVD1¥Installers¥Windows¥Agent¥SysMan¥CA_SystemEDGE_AdvancedEncryption
フォルダを開いて **ca-setup.exe** をダブルクリックします。

SystemEDGE Advanced Encryption のインストール ウィザードが表示されます。
2. [次へ] をクリックします。

[使用許諾契約書] ページが表示されます。
3. [使用許諾契約書に同意します] を選択し、[次へ] をクリックします。

[設定の確認] ページが表示されます。
4. [インストール] をクリックします。

インストールが完了すると、[インストールが完了しました] ページが表示されます。

インストールを確認するには、SystemEDGE のインストール ディレクトリに暗号化ライブラリが存在していることを確認します。詳細については、「インストールされるファイル」を参照してください。

Windows での SystemEDGE Advanced Encryption のアンインストール

SystemEDGE Advanced Encryption に依存するパッケージがインストールされていないことを確認します。以下の手順に従ってください。

Windows で SystemEDGE Advanced Encryption をアンインストールする方法

1. [プログラムの追加と削除] ダイアログ ボックスを開き、[SystemEDGE Advanced Encryption] を選択して [削除] をクリックします。

確認ダイアログ ボックスが開きます。
2. [はい] をクリックします。

パッケージがアンインストールされます。

UNIX での SystemEDGE Advanced Encryption のインストール

SystemEDGE Advanced Encryption インストールパッケージは、SystemEDGE によりサポートされるすべての UNIX プラットフォームおよびアーキテクチャで利用できます。使用しているシステムとアーキテクチャに適したパッケージをインストールします。

UNIX で SystemEDGE Advanced Encryption をインストールする方法

1. root ユーザとしてシステムにログインし、
DVD2¥Installers¥<platform>¥Agent¥SysMan¥CA_SystemEDGE_AdvancedEncryption フォルダのローカル コピーを作成します。このとき、使用しているオペレーティング システムに対応する *platform* フォルダを選択します。
2. 以下のように、このディレクトリからインストールを実行します。

```
./ca-setup.sh CA_SETUP_JRE_DIR=""
```

```
CA_SETUP_JRE_DIR
```

システムの **Java** ディレクトリを指定し、インストーラがインストール ウィザード ダイアログ ボックスを表示できるようにします。**Java** が利用可能でない場合、ダイアログ ボックスはテキスト モードのコンソールに表示されます。

インストールが完了すると、[インストールが完了しました] ページが表示されます。

インストールを確認するには、SystemEDGE のインストール ディレクトリに暗号化ライブラリが存在していることを確認します。詳細については、「インストールされるファイル」を参照してください。

UNIX での SystemEDGE Advanced Encryption のアンインストール

SystemEDGE Advanced Encryption に依存するパッケージがインストールされていないことを確認します。以下の手順に従ってください。

UNIX で Advanced Encryption をアンインストールする方法

1. ターミナル コンソールを開きます。
2. システムの CA_SystemEDGE_AdvancedEncryption フォルダに移動し、以下コマンドを実行します。

```
./ca-setup.sh -x
```

3. エージェントによりアンインストールされます。

インストールされるファイル

SystemEDGE Advanced Encryption をインストールすると、以下のファイルが Windows システムにインストールされます。

- インストールされた以下のファイルにより、認証および暗号化（非 FIPS モード）が提供されます。
 - libcapki.dll
 - libcapki_thread.dll
 - libcaopenssl_crypto.dll
 - libcaopenssl_ssl.dll
 - libcapki_ipthread.dll
- FIPS モードで実行できるようにするため、以下のファイルがインストールされます。
 - cryptocme2.dll
 - cryptocme2.sig
 - ccme_base.dll
 - ccme_ecc.dll
 - ccme_eccaccel.dll

詳細については、付録「FIPS 140-2 の暗号化」を参照してください。

SystemEDGE Advanced Encryption をインストールすると、以下のファイルが UNIX システムにインストールされます。

- インストールされた以下のファイルにより、認証および暗号化（非 FIPS モード）が提供されます。
 - libcapki
 - libcapki_thread_posix
 - libcaopenssl_crypto
 - libcaopenssl_ssl
 - libetpki_openssl_ssl

注: ファイル拡張子は、オペレーティング システムに応じて .so、.sl、または .dylib です。

- FIPS モードで実行できるようにするため、以下のファイルがインストールされます。

- libcryptocme2
- libcryptocme2.sig
- libccme_base
- libccme_ecc
- libccme_eccaccel

詳細については、付録「**FIPS 140-2** の暗号化」を参照してください。

注: ファイル拡張子は、オペレーティング システムに応じて **.so** または **.sl** (**.sig** ファイルを除く) です。

付録 C: 他の SNMP エージェントとの共存

この章では、他の SNMP エージェントと共に SystemEDGE エージェントを使用する方法について説明します。

このセクションには、以下のトピックが含まれています。

[複数の SNMP エージェントのサポート \(P. 529\)](#)

[複数の SystemEDGE インスタンスのサポート \(P. 530\)](#)

[Microsoft Windows SNMP エージェントとの共存 \(P. 531\)](#)

[HP SNMP エージェントとの共存 \(P. 531\)](#)

[AIX SNMP エージェントとの共存 \(P. 532\)](#)

複数の SNMP エージェントのサポート

デフォルトでは、SNMP エージェントは UDP ポート 161 を使用しようとしませんが、一度に特定のポートにバインドできるのは 1 つのプロセスだけです。そのため、一度に 1 つの SNMP エージェントのみが UDP ポート 161 を使用できます。複数の SNMP エージェントを同時に実行している場合、それらの共存戦略を作成する必要があります。

複数の SNMP エージェントが UDP ポート 161 を共有できるようにするために考えられるソリューションを以下に示します。

- 通常はマスタ/スレーブ関係にある複数の SNMP エージェント間で UDP ポート 161 を多重化します。
- 各エージェントが別個の UDP ポートにバインドされ、管理ソフトウェアがそれらと個別に通信するように設定される場合は、エージェントを同時に実行します。

注: UDP ポート 1691 は、SystemEDGE エージェントによる使用のために予約されており、CA Virtual Assurance からのエージェント展開のデフォルトポートです。UDP/161 の代わりにそのポートを使用するようにエージェントを設定できます。代替ポートでエージェントを起動する方法の詳細については、「エージェントの起動」の章を参照してください。

- 既存の独自マスタ/サブエージェント API すべてに従うエージェントをコーディングします。このソリューションには、コーディング、テスト、および多数の独自 API のライセンス取得の点で非常に高いコストがかかります。

エージェント多重化の最も一般的な概念には、別個のサブエージェントまたは別々の MIB を実装するスレーブエージェントと通信する UDP ポート 161 にバインドされたマスタエージェントが必要です。マスタとサブエージェントは、サブエージェントを登録し、それらの間でメッセージを交換するための定義済みプロトコルを使用して通信します。マスタおよびサブエージェント間でそのような通信プロトコルを制御するためのプロトコルはいくつか存在します。

SystemEDGE のインストール時、ネイティブのオペレーティングシステムエージェントを SystemEDGE エージェントに置き換えることもできます。エージェントを置き換える場合、ネイティブエージェントの設定を自動的に継承することを選択できます。インストール時にネイティブ SNMP エージェントを置き換える方法の詳細については、「インストールと展開」の章を参照してください。

複数の SystemEDGE インスタンスのサポート

公式にサポートされている機能ではありませんが、テストインスタンスでのテスト設定の変更など、1つのサーバでコマンドシェルから SystemEDGE エージェントの複数インスタンスを実行することができます。エージェントを起動するコマンドラインオプションでは、リスニングポートと設定ファイルの場所を指定できます。各インスタンスを開始するときに、コマンドラインで別々のポートを入力することにより、一度にエージェントの複数のインスタンスを開始できます。必要な場合は別々の設定ファイルを使用するように各エージェントを編集および設定できます。

重要: 複数の SystemEDGE インスタンスは、エージェントインストールまたは CA Virtual Assurance の展開および管理ではサポートされません。インストール時に指定したデフォルトポート上で実行されるインスタンスのみがサポートされます。エージェントをアンインストールする必要がある場合、デフォルトインスタンスのみが削除されます。同じサーバ上の追加のインスタンスはすべて手動で設定、実装、およびメンテナンスする必要があります。

1つのサーバ上で複数のエージェントインスタンスを実行する方法の詳細については、「エージェントの起動」の章を参照してください。

Microsoft Windows SNMP エージェントとの共存

SystemEDGE エージェントは、Microsoft SNMP エージェントのサブエージェントではなくなりました。Microsoft Windows 拡張エージェントと共存できます。

以下のいずれかの共存方法を採用します。

- ポート 1691 上で SystemEDGE エージェントを実行して、ポート 161 上で Microsoft エージェントを実行することで、システム上で Microsoft エージェントと SystemEDGE エージェントを同時に実行します。
- Microsoft マスタ エージェントをオフまたは無効にし、ポート 161 上で SystemEDGE エージェントを実行します。インストール時にネイティブ エージェントを置き換えると、この方法が採用されます。

HP SNMP エージェントとの共存

HP-UX システムには、MIB-II および HP プライベート企業 MIB をサポートする、拡張エージェントおよびサブエージェントが用意されています。マスタ/サブエージェント インターフェースは、SystemEDGE エージェントでサポートされない独自 API に基づいています。

以下のいずれかの共存方法を採用します。

- ポート UDP/1691 で SystemEDGE エージェントを実行することで、システム上で HP エージェントと SystemEDGE エージェントを同時に実行します。
- HP マスタまたはサブエージェントをオフまたは無効にし、ポート UDP/161 上で SystemEDGE エージェントを実行します。インストール時にネイティブ エージェントを置き換えると、この方法が採用されます。

AIX SNMP エージェントとの共存

AIX リリース 4.1 以降のシステムには、SMUX もサポートされる MIB-II SNMP エージェントが用意されています。AIX エージェントに加えて、または AIX エージェントの代わりに SystemEDGE エージェントを実行できます。

以下のいずれかの共存方法を採用します。

- SystemEDGE エージェントと AIX エージェントを同時に実行します。そのためには、AIX の TCP/IP 起動スクリプト `/etc/rc.tcpip` で `-p 1691` オプションを使用して SystemEDGE エージェントが呼び出されることを確認してください。
- AIX エージェントの代わりに SystemEDGE エージェントを実行します。そのためには、以下の行の先頭にシャープ記号 (#) を追加することで、`/etc/rc.tcpip` で AIX エージェントの呼び出しをコメント解除します。

```
# Start up the Simple Network Management Protocol (SNMP) daemon
# start /usr/sbin/snmpd "$src_running"
```

インストール時にネイティブ エージェントを置き換えても、この方法が採用されます。

付録 D: Host Resources MIB

この章では、IETF Host Resources MIB（RFC 1514）を介して利用できる管理情報について説明します。

このセクションには、以下のトピックが含まれています。

[Host Resources MIB の概要](#) (P. 534)

[ホスト リソース システム グループ](#) (P. 535)

[ホスト リソース ストレージ グループ](#) (P. 536)

[ホスト リソース デバイス グループ](#) (P. 536)

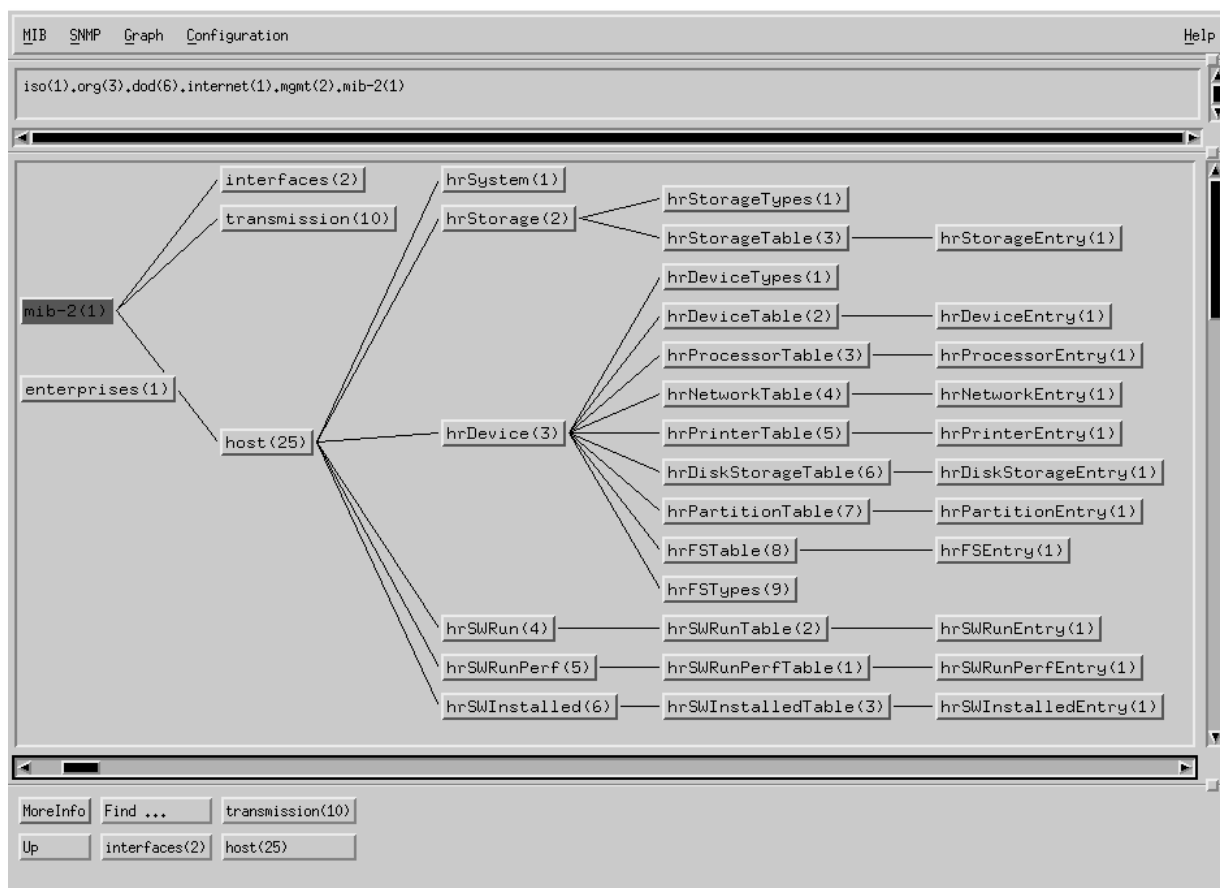
[ソフトウェア グループを実行するホスト リソース](#) (P. 541)

[ホスト リソースのインストール済みソフトウェア グループ](#) (P. 542)

[サポートされていない Host Resources MIB オブジェクト](#) (P. 543)

Host Resources MIB の概要

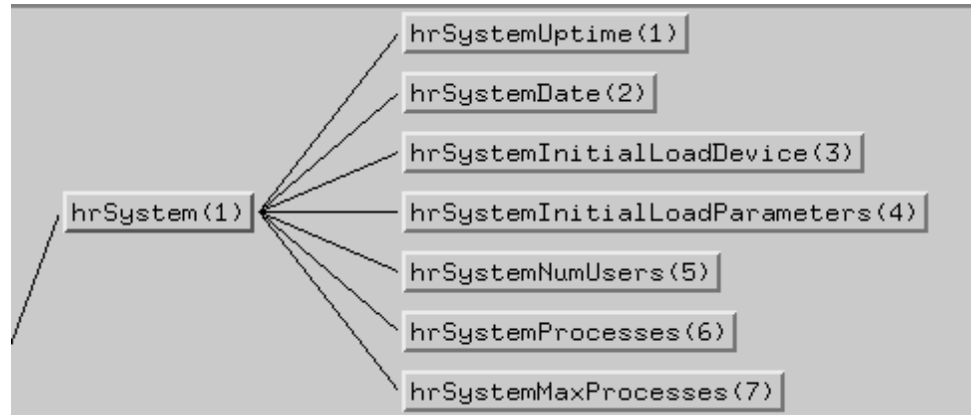
Host Resources MIB は、オペレーティング システム、ネットワーク サービス、どのソフトウェア アプリケーションからも独立した、ホスト コンピュータを管理できるオブジェクトのセットを定義します。Host Resources MIB 内に定義されたオブジェクトは、多くのコンピュータ システム アーキテクチャ間で共通です。以下の図に、Host Resources MIB の全体的な構成を示します。



注: この章の例では、Host Resources MIB 全体については説明しません。MIB 全体の説明については、エージェントインストールの `mib` サブディレクトリにある `hostmib.asn1` ファイルを参照してください。

ホスト リソース システム グループ

ホスト リソース システム (hrSystem) グループは、全体としてホスト システムに関連する情報を提供します。以下の図に、hrSystem グループの構成を示します。



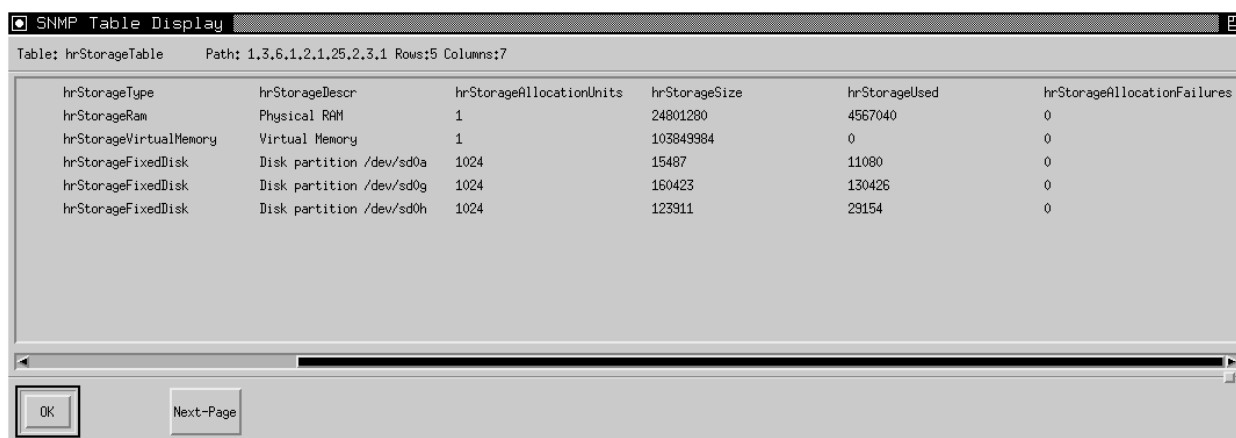
ホスト リソース システム グループを介して以下の情報が利用可能です。

- システムの稼働時間と日付
- オペレーティング システム ロード デバイス、パス名、およびパラメータ
- ホストが状態情報を格納しているユーザセッションの数
- 現在ロードまたは実行されているプロセスの数
- 実行できるプロセスの最大数

ホストリソース ストレージ グループ

ホストリソース ストレージ グループ (hrStorageTable) には、ホストシステムで割り当てられたストレージの論理的な領域がリストされます。これらのストレージ領域には、テープやフロッピー ドライブなどの物理ストレージの代わりに、アプリケーションが認識できるファイル システムおよびディスク パーティションが含まれます。

このテーブルは、メモリ不足およびバッファ不足タイプのエラーを診断するのに役立つように用意されています。さらに、メモリ、ディスク、またはバッファ使用状況を追跡するために役立つパフォーマンス モニタ ツールでもあります。以下の図に、ホストリソース ストレージ グループのサンプル出力を示します。



The image shows a window titled "SNMP Table Display" with a table of storage information. The table has 7 columns: hrStorageType, hrStorageDescr, hrStorageAllocationUnits, hrStorageSize, hrStorageUsed, and hrStorageAllocationFailures. The data is as follows:

hrStorageType	hrStorageDescr	hrStorageAllocationUnits	hrStorageSize	hrStorageUsed	hrStorageAllocationFailures
hrStorageRam	Physical RAM	1	24801280	4567040	0
hrStorageVirtualMemory	Virtual Memory	1	103849984	0	0
hrStorageFixedDisk	Disk partition /dev/sd0a	1024	15487	11080	0
hrStorageFixedDisk	Disk partition /dev/sd0g	1024	160423	130426	0
hrStorageFixedDisk	Disk partition /dev/sd0h	1024	123911	29154	0

ホストリソース デバイス グループ

ホストリソース デバイス (hrDevice) グループには、ホストシステム上のデバイスに関する情報を示す複数のテーブルが含まれています。このグループ内のメイン テーブルは hrDeviceTable です。ホストに存在するすべてのデバイスがリストされます。hrDevice グループには、特定のデバイス タイプのより詳細な情報を示す、多数のデバイス固有のテーブルも含まれています。

たとえば、このグループには、以下のデバイス固有のテーブルが含まれています。

- プロセッサ
- ネットワーク
- プリンタ
- ディスク ストレージ
- パーティション
- ファイル システム

ホストにプリンタ デバイスが含まれていると **hrDeviceTable** に示されている場合、たとえば、**hrPrinterTable** からプリンタに関するより詳細な情報を取得できます。以下のセクションでは、**hrDeviceTable** およびデバイス固有のテーブルについて説明します。

デバイス テーブル

hrDeviceTable には、ホストシステム上のデバイスがリスト表示されます。このテーブルには、各デバイスの以下の情報がリスト表示されます。

- デバイス タイプ
- デバイスの説明
- ステータス（たとえば、実行中またはダウン）
- デバイスで検出されたエラーの数

インベントリの追跡とアセット管理

hrDeviceTable は、インベントリの追跡およびアセット管理に特に役立ちます。ネットワーク上の各ワークステーションを手動で開いて検査し、イーサネットカード、シリアルデバイス、オーディオデバイス、ディスク ストレージデバイスなどの数およびタイプを調べる代わりに、一元管理システムを使用してネットワーク上の各システムから **hrDeviceTable** を取得できます。

プロセッサ テーブル

hrProcessorTable には、システムのプロセッサに関するデバイス固有の情報がリストされます。システムに搭載されているプロセッサが 1 つだけの場合、**hrProcessorTable** に含まれる行は 1 つだけです。**hrProcessorTable** の列には、プロセッサ（そのような ID が割り当てられており、プロセッサ ベンダーによって利用可能されている場合）に関連付けられたファームウェアの製品 ID とプロセッサ 負荷が示されます。負荷は、過去 1 分間のプロセッサがアイドルでない時間の割合の平均です。

プロセッサ ロードトラッキング

プロセッサ負荷の平均をモニタ（グラフ化）することで、プロセッサにかかる負荷を視覚的に追跡できます。たとえば、一定して高いレベルであることをグラフが示している（プロセッサの負荷が高かった示している）場合、ユーザとプロセスの数が多すぎるためホストシステムに負荷がかかりすぎているため、ユーザの数を制限したり、一部のプロセスを別のシステムで実行する必要があると判断できます。

プロセッサ負荷グラフが一時的に急上昇している場合、プロセッサを大量に消費するプロセスが実行されていることを示している可能性があります。そのようにプロセッサ負荷が大きい結果生じる問題を回避するため、プロセッサ負荷がそれほど大きくない時間にそのプロセスを実行するように再スケジュールすることができます。

ディスク ストレージ テーブル

hrDiskStorageTable には、ホストのディスク ストレージ デバイスに関する情報が表示されます。テーブルは、各ディスク ストレージ デバイスに対して以下をリスト表示します。

- インスタンス
- ディスク ストレージ デバイスが、読み書きを許可するか、または読み取り専用か
- ストレージ メディア タイプ（たとえば、ハードディスクまたはフロッピーディスク）
- ディスクがリムーバブルかどうか
- ストレージ容量（KB 単位）

パーティション テーブル

hrPartitionTable には、ディスク ストレージ デバイスに設定されているパーティションが表示されます。このテーブルには、各パーティションの以下の情報がリスト表示されます。

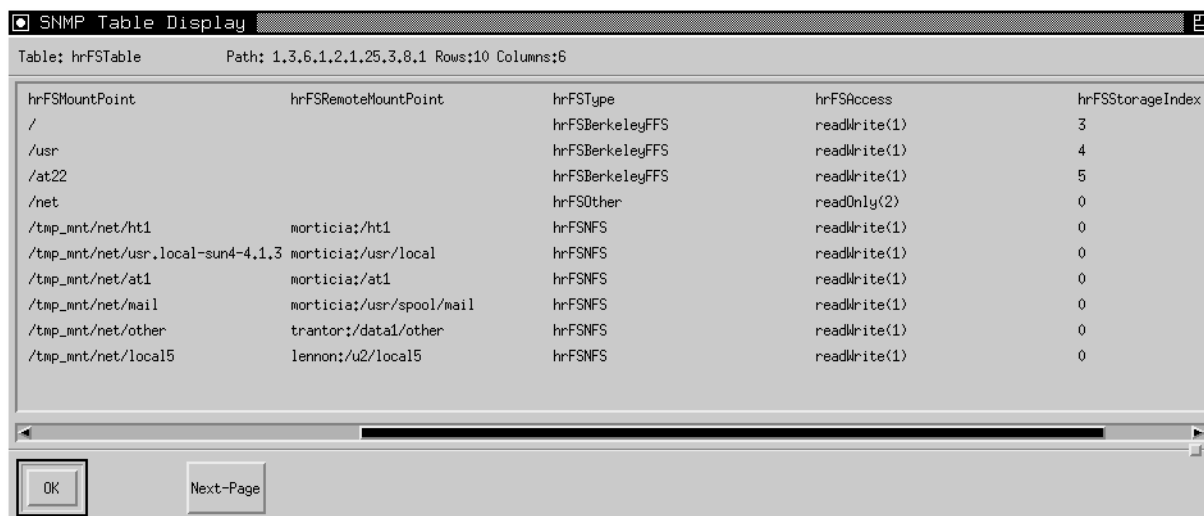
- インスタンス
- パーティション インデックス
- パーティション ラベル
- パーティション ID
- パーティション サイズ (KB 単位)
- そのパーティションにマウントされたファイル システムのインデックス

ファイル システム テーブル

hrFSTable には、ホストのローカルおよびリモートの両方のファイル システムに関する情報が示されます。このテーブルには、各ファイル システムの以下の内容がリストされます。

- ローカル マウント ポイント
- リモート マウント ポイント（ファイル システムがリモート サーバからマウントされている場合）
- ファイル システムのタイプ（たとえば、**BerkeleyFFS** など）
- ファイル システムで読み書きアクセスと読み取り専用アクセスのどちらが許可されるか

以下の図に、hrFSTable のサンプル出力を示します。



hrFSMountPoint	hrFSRemoteMountPoint	hrFSType	hrFSAccess	hrFSStorageIndex
/		hrFSBerkeleyFFS	readWrite(1)	3
/usr		hrFSBerkeleyFFS	readWrite(1)	4
/at22		hrFSBerkeleyFFS	readWrite(1)	5
/net		hrFSOther	readOnly(2)	0
/tmp_wnt/net/ht1	morticia:/ht1	hrFSNFS	readWrite(1)	0
/tmp_wnt/net/usr.local-sun4-4.1.3	morticia:/usr/local	hrFSNFS	readWrite(1)	0
/tmp_wnt/net/at1	morticia:/at1	hrFSNFS	readWrite(1)	0
/tmp_wnt/net/mail	morticia:/usr/spool/mail	hrFSNFS	readWrite(1)	0
/tmp_wnt/net/other	trantor:/data1/other	hrFSNFS	readWrite(1)	0
/tmp_wnt/net/local5	lennon:/u2/local5	hrFSNFS	readWrite(1)	0

ソフトウェア グループを実行するホストリソース

hrSWRun テーブルには、ホストシステムで現在実行されているソフトウェアがリスト表示されます。このテーブルには、実行中の各ソフトウェア プロセスの以下の情報がリスト表示されます。

- 一意の ID 番号
- 名前
- 製品 ID
- ソフトウェアが存在するディレクトリ パス
- ソフトウェア開始時のランタイム パラメータ
- ソフトウェアのタイプ（たとえば、オペレーティング システムまたはアプリケーション）
- ソフトウェアのステータス（たとえば、実行中、実行可能、実行不可能、または無効）

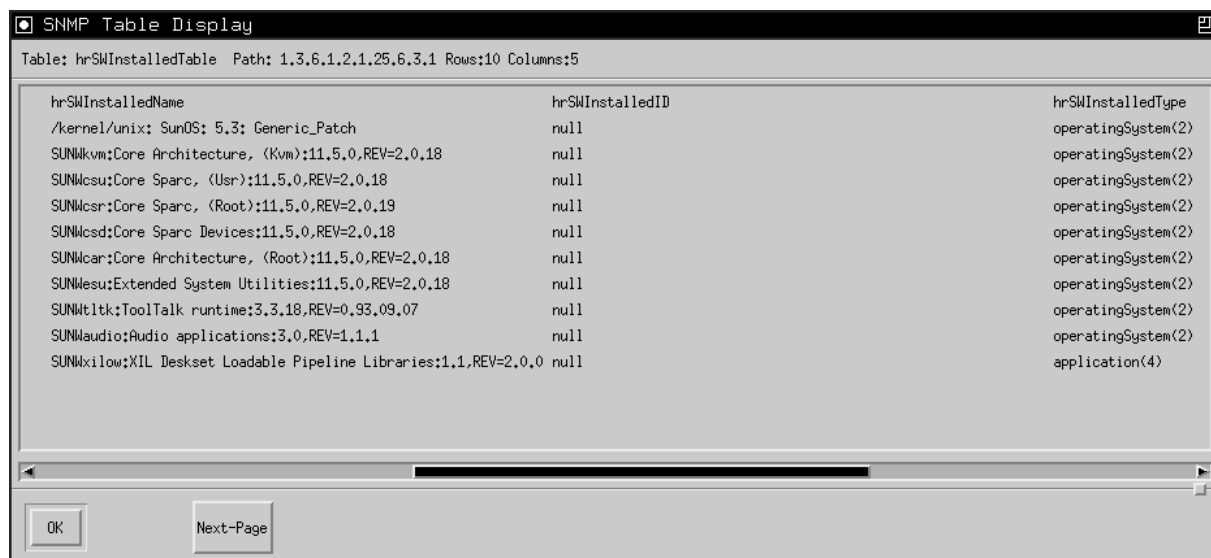
ホストリソースのインストール済みソフトウェア グループ

hrSWInstalled テーブルには、ホスト システムに現在インストールされているソフトウェアがリストされます。テーブルには、各ソフトウェア パッケージの以下の内容がリストされます。

- 一意の ID
- ソフトウェア パッケージ名
- 説明
- インストール日

このテーブルには、オペレーティング システムのパッチおよびインストールされたソフトウェアのバージョンが示されます。システム間でソフトウェア バージョンの一貫性を確認するのに適しています。以下の図に、サンプル

hrSWInstalledTable の一部を示します。



hrSWInstalledName	hrSWInstalledID	hrSWInstalledType
/kernel/unix: SunOS: 5.3: Generic_Patch	null	operatingSystem(2)
SUNWkvm:Core Architecture, (Kvm):11.5.0,REV=2.0.18	null	operatingSystem(2)
SUNWcsu:Core Sparc, (Usr):11.5.0,REV=2.0.18	null	operatingSystem(2)
SUNWcsr:Core Sparc, (Root):11.5.0,REV=2.0.19	null	operatingSystem(2)
SUNWcsd:Core Sparc Devices:11.5.0,REV=2.0.18	null	operatingSystem(2)
SUNWcar:Core Architecture, (Root):11.5.0,REV=2.0.18	null	operatingSystem(2)
SUNWesu:Extended System Utilities:11.5.0,REV=2.0.18	null	operatingSystem(2)
SUNWtltk:ToolTalk runtime:3.3.18,REV=0.93.09.07	null	operatingSystem(2)
SUNWaudio:Audio applications:3.0,REV=1.1.1	null	operatingSystem(2)
SUNWxilow:XIL Deskset Loadable Pipeline Libraries:1.1,REV=2.0.0	null	application(4)

注: システムによっては、このテーブルはサポートされません。オペレーティング システムでこのテーブルがサポートされるかどうかを調べるには、SystemEDGE のリリース ノートを参照してください。

サポートされていない Host Resources MIB オブジェクト

SystemEDGE エージェントは、サポートされているすべてのオペレーティングシステム上で **Host Resources MIB (RFC 1514)** をサポートしますが、この MIB モジュール内の MIB オブジェクトの一部はエージェントまたは基礎となるオペレーティングシステムによってサポートされません。このため、それらのオブジェクトは、影響を受けるオペレーティングシステムにインストールされたエージェントのバージョンによって実装できません。

すべてのオペレーティングシステムのサポートされない **Host Resources MIB** オブジェクトのリストについては、**SystemEDGE** のリリース ノートを参照してください。

付録 E: プライベート企業トラップ

SystemEDGE エージェントは、標準的な MIB-II トラップのサポートに加えて、エージェントのセルフ モニタリング機能で使用するために定義された、プライベート企業トラップ タイプの多くをサポートします。この付録では、SystemEDGE エージェントが送信できるトラップ PDU の形式について説明します。

このセクションには、以下のトピックが含まれています。

[トラップ PDU の形式](#) (P. 545)

[SNMPv1 トラップの形式](#) (P. 555)

トラップ PDU の形式

Systems Management の Empire MIB ファイル、`empire.asn1` は、SystemEDGE エージェントで使用する企業固有のトラップを定義します。このセクションでは、トラップのタイプごとの PDU 形式について、変数バインディングフィールド内の情報も含めて説明します。

変数バインディングは、トラップ PDU 内のオプションフィールドで、トラップに関連する重要な付加情報を提供します。

注: この章のテキストは、Systems Management の Empire MIB 定義から引用しています。この定義は、エージェントのインストールディレクトリ内の `mib` サブディレクトリ内にあります。

SystemEDGE エージェントによって送信されるすべてのトラップには、企業システムのオブジェクト識別子 (`sysObjectID`)、`empire(546).1.1` が含まれます。

monitor トラップ

monitor トラップの形式を次に示します。**monitor** トラップは、**SystemEDGE** エージェントが送信するトラップで、セルフ モニタ イベントが発生したことを示します。エージェントがこのトラップを送信するのは、モニタ対象の属性がしきい値に到達し、セルフ モニタの重大度が **none(1)** の場合です。それ以外の場合、エージェントは **aggregateState** トラップを送信します。

monitorEvent OBJECT IDENTIFIER ::= { traps 1 }

monitorTrap TRAP-TYPE

ENTERPRISE sysmgmt

VARIABLES { monDescr, monOID, monCurrVal, monValue, monRowStatus, monOperator, monIndex, monFlags, monObjClass, monObjInstance, monObjAttribute, monCurrState }

DESCRIPTION

「セルフ モニタのしきい値条件 (**monitorTable** 参照) に違反が発生しました。」

::= 1

MonitorEntryNotReady トラップ

MonitorEntryNotReady トラップの形式を次に示します。**MonitorEntryNotReady** トラップは、**SystemEDGE** エージェントが送信するトラップで、セルフ モニタ テーブルエントリの **monRowStatus** フィールドが **notReady(3)** に設定されていることを示します。エージェントがこのトラップを送信するのは、モニタの重大度が **none(1)** の場合のみです。

monitorEntryNotReadyEvent OBJECT IDENTIFIER ::= { traps 3 }

monitorEntryNotReadyTrap TRAP-TYPE

ENTERPRISE sysmgmt

VARIABLES { monDescr, monOID, monCurrVal, monValue, monRowStatus, monOperator, monIndex, monFlags, monObjClass, monObjInstance, monObjAttribute, monCurrState }

説明

「セルフ モニタの行ステータス (**monitorTable** 参照) が **notReady(3)** になりました。通常は、モニタ対象の **OID** が存在しないことが原因です。」

::= 3

logMonMatch トラップ

logMonMatch トラップの形式を次に示します。logMonMatch トラップは、SystemEDGE エージェントが送信するトラップで、エージェントが現在モニタ中のログファイル内に、ログファイルモニタリングサブシステムによって一致が検出されたことを示します。モニタ重大度は、変数バインディング内に含まれています。

```
logMonMatchEvent OBJECT IDENTIFIER ::= { traps 4 }
```

```
logMonMatchTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { logMonitorLogFile, logMonitorRegularExpression,
logMonitorLastTrap, logMonitorLastMatch,
logMonitorDescr, logMonitorIndex, logMonitorFlags,
logMonitorLogFileName, logMonitorSeverity }
```

DESCRIPTION

「ログ ファイル モニタの正規表現パターン (logMonitorTable 参照) が一致しました。」

```
::= 4
```

logMonNotReady トラップ

logMonNotReady トラップの形式を次に示します。logMonNotReady トラップは、SystemEDGE エージェントが送信するトラップで、ログ モニタ テーブルのエントリのステータスが notReady に変わったことを示します。

```
logMonNotReadyEvent OBJECT IDENTIFIER ::= { traps 5 }
```

```
logMonNotReadyTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { logMonitorLogFile, logMonitorRegularExpression,
logMonitorLastTrap, logMonitorLastMatch,
logMonitorDescr, logMonitorIndex, logMonitorFlags,
logMonitorLogFileName, logMonitorSeverity }
```

DESCRIPTION

「ログ ファイル モニタの行ステータス (logMonitorTable 参照) が notReady(3) になりました。通常は、ログ ファイルが存在しないことが原因です。」

```
::= 5
```

ntEventMonMatch トラップ

ntEventMonMatch トラップの形式を次に示します。ntEventMonMatch トラップは、SystemEDGE エージェントが送信するトラップで、エージェントが現在モニタ中の Windows イベント ログ ファイル内に、イベント ログ モニタ サブシステムによって一致が検出されたことを示します。モニタ重大度は、変数バインディング内に含まれています。

```
ntEventMonMatchEvent OBJECT IDENTIFIER ::= { traps 7 }
```

```
ntEventMonMatchTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { ntEventMonLog, ntEventMonTypeLastMatch,
ntEventMonTime, ntEventMonSrcLastMatch,
ntEventMonDescLastMatch, ntEventMonDescr,
ntEventMonIndex, ntEventMonFlags, ntEventMonSeverity }
```

DESCRIPTION

「NT イベント ログ モニタのすべてのフィルタ (log、type、src、desc。ntEventMonTable 参照) が一致しました。」

```
::= 7
```

ntEventMonNotReady トラップ

ntEventMonNotReady トラップの形式を次に示します。ntEventMonNotReady トラップは、SystemEDGE エージェントが送信するトラップで、Windows イベント ログ モニタのエントリのステータスが notReady(3) になったことを示します。

```
ntEventMonNotReadyEvent OBJECT IDENTIFIER ::= { traps 8 }
```

```
ntEventMonNotReadyTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { ntEventMonLog, ntEventMonTypeFilter,
ntEventMonSrcFilter, ntEventMonDescFilter,
ntEventMonDescr, ntEventMonIndex, ntEventMonFlags, ntEventMonSeverity }
```

DESCRIPTION

「Windows イベント ログ モニタの行ステータス (ntEventMonTable 参照) が notReady(3) になりました。通常は、イベント ログが存在しないことが原因です。」

```
::= 8
```

monitorClear トラップ

monitorClear トラップの形式を次に示します。**monitorClear** トラップは、**SystemEDGE** エージェントが送信するトラップで、以前存在していた、しきい値違反の条件が、存在しなくなったことを示します。**SystemEDGE** エージェントがこのトラップを送信するのは、クリア トラップの送信フラグを持つセルフ モニタ テーブルのエントリが **True** から **False** に移行した場合です。エージェントがこのトラップを送信するのは、モニタの重大度が **none(1)** の場合のみです。それ以外の場合、エージェントは **aggregateState** トラップを送信します。

monitorClearEvent OBJECT IDENTIFIER ::= { traps 9 }

monitorClearTrap TRAP-TYPE

ENTERPRISE sysmgmt

VARIABLES { monDescr, monOID, monCurrVal, monValue, monRowStatus, monOperator, monIndex, monFlags, monObjClass, monObjInstance, monObjAttribute, monCurrState }

説明

「セルフ モニタのしきい値条件 (**monitorTable** 参照) への違反がなくなりました。」

::= 9

processStop トラップ

processStop トラップの形式を次に示します。**processStop** トラップは、**SystemEDGE** エージェントが送信するトラップで、エージェントがモニタしていたプロセスの実行が停止されたか、実行できない状態にあることを示します。エージェントがこのトラップを送信するのは、プロセス モニタの重大度が **none(1)** の場合のみです。それ以外の場合、エージェントは **aggregateState** トラップを送信します。

processStopEvent OBJECT IDENTIFIER ::= { traps 10 }

processStopTrap TRAP-TYPE

ENTERPRISE sysmgmt

VARIABLES { pmonIndex, pmonDescr, pmonAttribute, pmonCurrVal, pmonOperator, pmonValue, pmonFlags, pmonRegExpr, pmonCurrentPID, pmonObjClass, pmonObjInstance, pmonObjAttribute, pmonCurrState, pmonProcName }

DESCRIPTION

「**procAlive** を使用してプロセス モニタによってモニタされていたプロセス (または **Windows** サービス、**processMonTable** 参照) が実行されていません。」

::= 10

processStart トラップ

processStart トラップの形式を次に示します。**processStart** トラップは、**SystemEDGE** エージェントが送信するトラップで、停止されていたプロセスが再起動されたこと（そして、**SystemEDGE** エージェントによって再取得されたこと）を示します。エージェントがこのトラップを送信するのは、プロセス モニタの重大度が **none(1)** の場合のみです。それ以外の場合、エージェントは **aggregateState** トラップを送信します。

```
processStartEvent OBJECT IDENTIFIER ::= { traps 11 }
```

```
processStartTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { pmonIndex, pmonDescr, pmonAttribute, pmonCurrVal, pmonOperator,
pmonValue, pmonFlags, pmonRegExpr, pmonCurrentPID, pmonObjClass, pmonObjInstance,
pmonObjAttribute, pmonCurrState, pmonProcName}
```

DESCRIPTION

「**procAlive** を使用してプロセス モニタによってモニタされていたプロセス（または **Windows** サービス、**processMonTable** 参照）が再起動されました。」

```
::= 11
```

processThreshold トラップ

processThreshold トラップの形式を次に示します。**processThreshold** トラップは、**SystemEDGE** エージェントが送信するトラップで、プロセスのモニタ対象属性が、指定されたしきい値条件に違反したことを示します。エージェントがこのトラップを送信するのは、プロセス モニタの重大度が **none(1)** の場合のみです。それ以外の場合、エージェントは **aggregateState** トラップを送信します。

```
processThresholdEvent OBJECT IDENTIFIER ::= { traps 12 }
```

```
processThresholdTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { pmonIndex, pmonDescr, pmonAttribute, pmonCurrVal, pmonOperator,
pmonValue, pmonFlags, pmonRegExpr, pmonCurrentPID, pmonObjClass, pmonObjInstance,
pmonObjAttribute, pmonCurrState, pmonProcName}
```

DESCRIPTION

「プロセス モニタのしきい値条件（**processMonTable** 参照）に違反が発生しました。」

```
::= 12
```

processClear トラップ

processClear トラップの形式を次に示します。processClear トラップは、SystemEDGE エージェントが送信するトラップで、エージェントが以前 processThreshold トラップを送信したプロセスの属性が、しきい値レベルではなくなったことを示します。エージェントがこのトラップを送信するのは、プロセスモニタの重大度が **none(1)** の場合のみです。それ以外の場合、エージェントは aggregateState トラップを送信します。

```
processClearEvent OBJECT IDENTIFIER ::= { traps 13 }
```

```
processClearTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { pmonIndex, pmonDescr, pmonAttribute, pmonCurrVal, pmonOperator,
pmonValue, pmonFlags, pmonRegExpr, pmonCurrentPID, pmonObjClass, pmonObjInstance,
pmonObjAttribute, pmonCurrState, pmonProcName}
```

DESCRIPTION

「プロセス モニタのしきい値条件 (processMonTable 参照) への違反がなくなりました。」

```
::= 13
```

processNotReady トラップ

processNotReady トラップの形式を次に示します。processNotReady トラップは、SystemEDGE エージェントが送信するトラップで、プロセス モニタ テーブル エントリの行ステータスが notReady(3) に設定されていることを示します。プロセスが procAlive を使用してモニタされている場合、代わりに processStop トラップが送信されます。エージェントがこのトラップを送信するのは、プロセス モニタの重大度が none(1) の場合のみです。

```
processNotReadyEvent OBJECT IDENTIFIER ::= { traps 14 }
```

```
processNotReady TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { pmonIndex, pmonDescr, pmonAttribute, pmonCurrVal, pmonOperator,
pmonValue, pmonFlags, pmonRegExpr, pmonCurrentPID, pmonObjClass, pmonObjInstance,
pmonObjAttribute, pmonCurrState, pmonProcName}
```

```
DESCRIPTION
```

「procAlive を使用していないプロセス モニタの行ステータス（processMonTable 参照）が notReady(3) になりました。通常は、プロセスが存在しないことが原因です。」

```
::= 14
```

addrChange トラップ

addrChange トラップの形式を次に示します。addrChange トラップは、SystemEDGE エージェントが送信するトラップで、基盤の IP アドレスが変更されたことを示します。

```
addrChangeEvent OBJECT IDENTIFIER ::= { traps 18 }
```

```
addrChangeTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { nodename, sysedgeAddressList }
```

```
説明
```

「エージェントのマシンの IP アドレスが変更されました。通常は、DHCP などの管理手段によって変更されています。」

```
::= 18
```


procGroupChange トラップ

procGroupChange トラップの形式を次に示します。procGroupChange トラップは、SystemEDGE エージェントが送信するトラップで、プロセス グループが変更されたことを示します。 モニタ重大度は、変数バインディング内に含まれています。

```
procGroupChangeEvent OBJECT IDENTIFIER ::= { traps 19 }
```

```
procGroupChangeTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { pgmonIndex, pgmonDescr, pgmonFlags, pgmonNumProcs, pgmonProcRegExpr,
pgmonRowStatus, pgmonPIDList, pgmonStatusList, pgmonSeverity, pgmonProcNameList }
```

説明

「プロセス グループ モニタのメンバシップ (processGroupMonTable 参照) が変更されました。通常は、プロセスが結合されたか、グループから削除されたプロセスがあることが原因です。」

```
::= 19
```

aggregateState トラップ

aggregateState トラップの形式を次に示します。aggregateState トラップは、SystemEDGE エージェントが送信するトラップで、集計の状態が変更されたことを示します。 エージェントがこのトラップを送信するのは、モニタのしきい値違反によって、集計の状態が変わる場合のみです。たとえば、重大度が「重大」のモニタで違反が発生したものの、集計の現在の状態が「致命的」の場合、集計の状態に影響はないため、エージェントはトラップを送信しません。

```
aggregateStateEvent OBJECT IDENTIFIER ::= {traps 20}
```

```
aggregateStateTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { aggIndex, aggObjClass, aggObjInstance, aggObjAttribute, aggCurrState,
aggPrevState, aggMonTable, aggMonIndex, aggMonCurrVal, aggMonOperator, aggMonValue }
```

説明

「集計の状態 (aggregateTable 参照) が変わりました。」

```
::= 20
```

aggregateAdd トラップ

aggregateAdd トラップの形式を次に示します。aggregateAdd トラップは、SystemEDGE エージェントが送信するトラップで、集計が集計テーブルに追加されたことを示します。

```
aggregateAddEvent OBJECT IDENTIFIER ::= {traps 21}
```

```
aggregateAddTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { aggIndex, aggObjClass, aggObjInstance, aggObjAttribute }
```

説明

「新しい集計 (aggregateTable 参照) が追加されました。」

```
::= 21
```

aggregateDelete トラップ

aggregateDelete トラップの形式を次に示します。aggregateDelete トラップは、SystemEDGE エージェントが送信するトラップで、集計が集計テーブルから削除されたことを示します。

```
aggregateDeleteEvent OBJECT IDENTIFIER ::= {traps 22}
```

```
aggregateDeleteTrap TRAP-TYPE
```

```
ENTERPRISE sysmgmt
```

```
VARIABLES { aggIndex, aggObjClass, aggObjInstance, aggObjAttribute }
```

説明

「集計 (aggregateTable 参照) が削除されました。」

```
::= 22
```

SNMPv1 トラップの形式

SystemEDGE エージェントはトラップ PDU を SNMPv1 トラップ ポート (UDP/162) に送信します。SNMPv1 トラップ PDU のコンポーネントを以下に示します。

エンタープライズ

送信者のシステム オブジェクト ID を指定します (たとえば、**empire(546).1.1**)。

エージェント アドレス

送信側ホストの IP アドレスを指定します。

汎用トラップ タイプ

以下のいずれかの汎用トラップ タイプを指定します。

- coldStart(0)
- warmStart(1)
- linkDown(2)
- linkup(3)
- authenFailure(4)
- egpNeighborloss(5)
- enterpriseSpecific(6)

固有のトラップ タイプ

企業固有のトラップ タイプを指定します。

タイム スタンプ

トラップが送信されたときの、**sysUptime** の値を指定します。**sendtrap** については、値は常に **0** です。

変数バインド

MIB 変数およびそれらの値の配列を指定します。

索引

6

64 ビット版 Linux のリリースでの
SystemEDGE インストールが失敗する - 128

A

addrChange トラップ - 597
aggregateAdd トラップ - 599
aggregateDelete トラップ - 599
aggregateState トラップ - 598
AIM と CA Virtual Assurance の統合 - 96
AIM アーキテクチャ - 94
AIM プラグインの検証 - 553
AIX SNMP エージェントとの共存 - 574
AIX システムのディスク パフォーマンス統
計収集の設定 - 246
AIX 論理パーティション グループ - 237
Application Insight Module (AIM) - 36

C

CA eHealth - 41
CA NSM - 42
CA Server Automation - 39
CA Spectrum - 40
CA SystemEDGE - 17
CA SystemEDGE : xtrapmon.exe ユーティリテ
ィは edgetrapmon.exe に置き換え - 560
CA Systems Management Empire MIB - 30
CA Technologies 製品リファレンス - 3
CA Virtual Assurance - 38
CA Virtual Assurance セキュリティ オプショ
ン - 93
CA Virtual Assurance との統合 - 90
CA Virtual Assurance によるエージェントの
展開 - 142
CA Virtual Assurance を使用したインストー
ル - 103
CA Virtual Assurance を使用した設定 - 158
CA への連絡先 - 3

CPU 統計テーブル - 247

D

diagsysedge.exe の使用 - 551
diagsysedge によるエージェントが実行され
ているかどうかの確認 - 552

E

edgemon の例 - 306
edgemon 自動ウォッチャーの例 - 386
edgemon の例 - 366, 422, 452, 488
edgemon ユーティリティ -- Windows イベ
ントのモニタ - 480
edgemon ユーティリティ -- 自動ウォッチ
ャーの使用 - 377
edgemon ユーティリティ -- プロセス グル
ープのモニタ - 416
edgemon ユーティリティ -- ログ ファイル
のモニタ - 446
emphistory ディレクティブ -- 履歴制御テー
ブルへのエントリの追加 - 502
emphistory の例 - 506
empire.asn1 で ntRegPerf 変数に関する編集
を行う方法 - 529
empire.asn1 で拡張変数に関する編集を行う
方法 - 519

F

FIPS 140-2 の暗号化 - 561
FIPS 140-2 モード - 561
FIPS 140-2 モードの設定 - 563
FIPS の互換性 - 566
FIPS モードの考慮事項 - 564
FIPS ライブラリのインストール - 561

G

GetBulk 応答メッセージサイズの設定 - 174

H

Host Resources MIB - 33, 575
Host Resources MIB の概要 - 576
HP SNMP エージェントとの共存 - 574

I

I/O バッファ キャッシュ グループ - 236
IPC メカニズムの削除 - 235
IP-MIB テーブル - 33

L

Linux の空きメモリ計算の設定 - 184
logMonMatch トラップ - 589
logMonNotReady トラップ - 589
LPAR グループ - 249

M

MIB-II - 32
MIB 拡張 - 34
MIB テーブルの制限の設定 - 180
Microsoft Windows SNMP エージェントとの共存 - 573
monitorClear トラップ - 592
MonitorEntryNotReady トラップ - 588
monitor ディレクティブ -- セルフ モニタ テーブルへのエントリの追加 - 277
monitor トラップ - 588

N

NFS 統計グループ - 244
ntEventMonMatch トラップ - 590
ntEventMonNotReady トラップ - 591
ntregperf キーワード -- Windows レジストリとパフォーマンス MIB オブジェクトの追加 - 527
ntRegPerf 変数に関する別個の MIB 仕様を追加する方法 - 530
NT イベント モニタ グループ - 241
NT イベント モニタ テーブル - 462

NT イベント モニタ テーブルのアクションパラメータ - 472
NT イベント モニタ テーブルの表示 - 473
NT イベント モニタ テーブルのフラグ - 469
NT イベント モニタ テーブルの列 - 463
NT イベント ログ モニタリング テーブル - 242
NT キャッシュ パフォーマンス グループ - 240
NT サービス テーブル - 239
NT システム グループ - 238
NT システム パフォーマンス グループ - 240
NT スレッド テーブル - 238
NT パフォーマンス グループ - 239
NT ページファイル パフォーマンス グループ - 241
NT メモリ パフォーマンス グループ - 240
NT レジストリ/パフォーマンス拡張グループ - 242
NT レジストリ グループ - 239

O

OID の逆ルックアップ - 281

P

Perfcube モジュールの設定 - 191
Perl 互換正規表現 (PCRE) のサポート - 195
processClear トラップ - 596
processNotReady トラップ - 597
processStart トラップ - 594
processStop トラップ - 593
processThreshold トラップ - 595
procGroupChange トラップ - 598

R

RPC 統計グループ - 243

S

SNMP SET 制限の設定 - 173
SNMP SET 通知の設定 - 172

SNMP User Datagram Protocol (UDP) 通信用
の IP ファミリの設定 - 161
SNMPv1 および SNMPv2c の無効化 - 214
SNMPv1 コミュニティのアクセス リスト -
167
SNMPv1 と SNMPv2c のコミュニティ - 46
SNMPv1 トラップ ソースの指定 - 168
SNMPv1 トラップ デスティネーションの設
定 - 168
SNMPv1 トラップの形式 - 600
SNMPv2c および SNMPv3 のトラップ デス
ティネーションの設定 - 211
SNMPv2 の行ステータス - 54
SNMPv3 エンジン ID の設定 - 203
SNMPv3 設定 - 202
SNMPv3 設定ファイルの暗号化 - 214
SNMPv3 のユーザとセキュリティ - 47
SNMPv3 ユーザ情報の設定 - 204
SNMPv3 ユーザのアドレス フィルタリング -
207
SNMP 情報の設定 - 160
SNMP テーブル - 48
SNMP トラップ - 48
SNMP のバージョンとアクセス - 45
SNMP バインドアドレスの設定 - 162
SNMP を介した拡張スクリプトの再設定 -
543
Solaris ゾーン プロセス グループ モニタリ
ング - 424
Solaris ゾーン プロセスのモニタリング - 371
sysedge.cf での自動ウォッチャーの指定 -
387
sysedge.cf と sysedge.mon のやり取り - 158
SystemEDGE 4.3.4 からのエージェント ア
ップグレード - 151
SystemEDGE Advanced Encryption - 565
SystemEDGE Advanced Encryption のインスト
ール - 566
SystemEDGE AIM - 531
SystemEDGE AIM のロード - 189
SystemEDGE エージェントの使用のガイドラ
イン - 97

SystemEDGE から送信されるトラップのエー
ジェント アドレス - 171
SystemEDGE コマンドライン ユーティリテ
ィ - 536
SystemEDGE コマンドライン ユーティリテ
ィの設定 - 537
SystemEDGE のその他のコマンドライン ユ
ーティリティ - 537
SystemEDGE のアンインストール - 144
Systems Management Empire MIB - 225
Systems Management Empire MIB 拡張グルー
プ - 509
Systems Management Empire MIB グループ -
62
Systems Management Empire MIB の ntRegPerf
グループ - 522
Systems Management Empire MIB の概要 - 226

U

UNIX システム上の SystemEDGE および AIM
のアンインストール - 146
UNIX および Linux システムでのインストー
ル - 120
UNIX システムおよび LINUX システムでのエ
ージェントのインストール - 120
UNIX システムでのエージェントの起動また
は停止 - 218
UNIX システムでのエージェントの自動起動
- 222
UNIX システム用のサービス起動スクリプト
- 218
UNIX での SystemEDGE Advanced Encryption
のアンインストール - 569
UNIX での SystemEDGE Advanced Encryption
のインストール - 568

W

watch logfile ディレクティブ -- ログ モニタ
テーブルへのエントリの追加 - 441
watch ntevent ディレクティブ -- NT イベント
モニタ テーブルへのエントリの追加 - 476

watch ntsservice ディレクティブ -- プロセス
モニタ テーブルへのサービス モニタリン
グ エントリの追加 - 350

watch process ディレクティブ -- プロセス モ
ニタ テーブルへのエントリの追加 - 346

watch procgroupex ディレクティブ -- [プロセ
ス グループ モニタ] テーブルへのエント
リの追加 (UNIX のみ) - 412

watch procgroup ディレクティブ -- [プロセス
グループ モニタ] テーブルへのエントリの
追加 - 410

Who テーブル - 231

Windows 上の SystemEDGE および AIM のア
ンインストール - 144

Windows イベントの検索条件 - 461

Windows イベント モニタのトラップ - 75

Windows イベント モニタリング - 74, 459

Windows イベント モニタリング エントリの
削除 - 491

Windows イベント モニタリングの概要 - 460

Windows イベント モニタリングの設定 - 474

Windows イベント モニタリングの例 - 479

Windows イベント モニタリング用の
edgewatch コマンド - 486

Windows サービスのモニタリング - 313

Windows システムでのインストール - 104

Windows システムでのエージェントの起動
または停止 - 217

Windows システムでのエージェントの自動
起動 - 222

Windows タスク スケジューラのモニタ - 539

Windows での SystemEDGE Advanced
Encryption のアンインストール - 568

Windows での SystemEDGE Advanced
Encryption のインストール - 567

Windows でのエージェントのインストール -
105

Windows の SystemEDGE コントロール パネ
ル - 215

Windows レジストリおよびパフォーマンス
機能 - 521

Windows レジストリおよびパフォーマンス
変数の設定 - 526

Windows レジストリとパフォーマンス MIB
オブジェクト - 521

Windows レジストリとパフォーマンスの例 -
528

あ

アーキテクチャの概要 - 44

アクションのサポートの設定 - 179

アクセス コミュニティの設定 - 165

イベント エンジン - 92

インスタンスによるモニタの作成 - 50

インストールされるファイル - 569

インストール時のエージェントの設定 - 103

インストールと展開 - 101

インストールに関する注意事項 - 102

インストールの前提条件 - 566

インベントリの追跡とアセット管理 - 579

ウォーム スタート ディスカバリの設定 -
184

ウォーム スタートの例外 - 223

エージェントが SNMP リクエストに応答し
ない - 555

エージェントが実行中であることの確認 -
556

エージェントがトラップを送信することの
確認 - 559

エージェント設定の確認 - 558

エージェントと AIM のアップグレード - 149

エージェントのウォーム スタート - 222

エージェントの起動 - 217

エージェントの設定 - 155

エージェントの展開 - 140

エージェントのトラップ設定の確認 - 559

応答ファイルの設定と使用 - 138

オブジェクト集計 - 81

オブジェクト ステータス - 79

オンデマンド モジュール - 95

か

カーネル設定グループ - 229
拡張キーワード -- 拡張グループへのエントリの追加 - 512
拡張グループ - 248
拡張グループ機能 - 510
拡張スクリプト - 516
拡張の使用に関する推奨事項 - 520
拡張の例 - 514
拡張変数の設定 - 511
隔離ファイルの存在のモニタ - 541
カスタム MIB オブジェクト - 509
管理システムが SNMP トラップ メッセージを受信しない - 559
管理システム設定の確認 - 558
管理システムのトラップ設定の確認 - 560
管理ソフトウェアでの Windows レジストリおよびパフォーマンス変数の使用 - 529
管理ソフトウェアでの拡張変数の使用 - 519
管理対象オブジェクトの作成 - 253, 315
管理対象ノードおよび AIM サーバのアップグレード - 149
関連ドキュメント - 18
キー保護 - 564
規則 - 19
起動時の初期設定 - 501
機能および統合概要 - 21
機能概要 - 21
クエリに対するエージェントの応答の確認 - 557
行作成 - 267
行作成オブジェクト - 331, 403, 433, 467, 498
グループテーブル - 230
コア AIM - 531
コマンドラインを使用した Windows へのエージェントのインストール - 112
コマンドライン起動オプション - 220
コマンドラインユーティリティ - 535
コマンドラインを使用した UNIX へのエージェントのインストール - 128
コンセプト - 43

さ

サービス自動ウォッチャーの例 - 393
再インストール - 148
サブプログラムのユーザおよびグループ権限の設定 (UNIX のみ) - 180
サポートされていない Host Resources MIB オブジェクト - 585
サポートされていない Systems Management Empire MIB オブジェクト - 250
サポートされている MIB - 29
サポートされている MIBs - 61
サポートされないパフォーマンス データ タイプ - 525
サポートされる暗号化プロトコル - 563, 566
サポートされる認証プロトコル - 563, 566
サポートされるプラットフォーム - 565
サンプルスクリプトのテスト - 517
サンプルセルフ モニタ テーブル エントリ - 265
サンプルプロセス モニタ テーブル エントリ - 327
システム グループ - 228
システム情報の設定 - 160
システムの初期化時におけるエージェントの起動の確認 - 556
システム ブート時のエージェントの自動起動 - 221
自動ウォッチャー - 373
自動ウォッチャー テーブル - 250
自動ウォッチャーの仕組み - 373
自動ウォッチャー用の edgewatch コマンド - 381
集計状態テーブル - 249, 256, 317
修正処置 - 35
循環ログ ファイル - 457
状態管理 - 28
状態管理と集計の仕組み - 84
状態管理トラップ - 87
状態管理の使用例 - 539
状態管理の設定オプション - 88
状態管理モデル - 77

診断ロギング メカニズムの設定 - 176
スーパー集計 - 82
ステートフル モニタリングの仕組み - 56
ステートレス モニタリングの仕組み - 58
ストリーム グループ - 229
ストリーム バッファ グループ - 236
ストリーム バッファ 割り当てテーブル - 236
正規表現の例 - 198
セキュリティの設定に関する推奨事項 - 192
設定の基本 - 155
設定ファイル - 51
セルフ モニタ テーブル - 243, 256
セルフ モニタ テーブルのアクション パラ
メータ - 271
セルフ モニタ テーブルの表示 - 274
セルフ モニタ テーブルのフラグ - 268
セルフ モニタ テーブルの列 - 258
セルフ モニタの状態管理 - 252
セルフ モニタのトラップ - 66
セルフ モニタリング - 60, 251
セルフ モニタリング エントリの削除 - 308
セルフ モニタリングの概要 - 251
セルフ モニタリングの設定 - 275
セルフ モニタリングの例 - 283
セルフ モニタリング用の edgemon コマンド
- 302
操作時の動的な設定 - 502
ソフトウェア グループを実行するホスト リ
ソース - 583

た

対象読者 - 17
多層階層でのモニタ集計の設定 - 185
他の SNMP エージェントとの共存 - 571
追加パラメータ - 513
ディスク ストレージ テーブル - 580
ディスク 統計テーブル - 246
ディレクトリ名のルックアップ キャッシュ
グループ - 236
ディレクトリ モニタリング - 428
テキスト パターン除外の設定 - 198

デバイス ステータス確認の制限の設定 - 182
デバイス テーブル - 228, 579
トップ プロセス AIM - 531
トラップ PDU の形式 - 587
トラップ コミュニティ テーブル - 237
トラブルシューティングと使用方法に関す
る推奨事項 - 551
トラブルシューティング用の診断ログ情報 -
555
トラブルシューティング用のレポートの取
得 - 553

な

認証失敗トラップの設定 - 172

は

パーティション テーブル - 581
バインド アドレスの設定 - 162
バインド失敗：アドレスがすでに使用中 -
560
はじめに - 17
バッファ グループ - 235
パフォーマンス キューブ AIM - 533
パフォーマンス キューブ AIM の有効化 -
534
パフォーマンス グループ - 233
パフォーマンス データ - 523
汎用自動ウォッチャー - 375
汎用自動ウォッチャーの例 - 392
ファイル システム テーブル - 582
ファイル システムのモニタ - 541
ファイル システム領域のモニタリング - 228
ファイル ベースの設定 - 159
ブート設定グループ - 229
複数行の正規表現 - 197
複数の SNMP エージェントのサポート - 571
複数の SystemEDGE インスタンスのサポート
- 573
プライベート企業トラップ - 587
プラグインと統合 - 36
プラットフォーム管理モジュール - 91

プラットフォーム サポート - 562
プロセス間通信グループ - 234
[プロセス グループ モニタ] テーブル - 249, 396
[プロセス グループ モニタ] テーブルのアクション パラメータ - 406
[プロセス グループ モニタ] テーブルの表示 - 408
プロセス グループ モニタ テーブルのフラグ - 404
[プロセス グループ モニタ] テーブルの列 - 397
プロセス グループ モニタのトラップ - 72
プロセス グループ モニタリング - 71, 395
プロセス グループ モニタリング エントリの削除 - 423
プロセス グループ モニタリングの概要 - 395
プロセス グループ モニタリングの設定 - 409
プロセス グループ モニタリングの例 - 414
プロセス グループ モニタリング用の edgewatch コマンド - 420
プロセス自動ウォッチャーの例 - 393
プロセス属性 - 329
プロセス テーブル - 230
プロセスとサービスの自動ウォッチャー - 376
プロセスとサービスのモニタリング - 68, 311
プロセスとサービスのモニタリングの概要 - 312
プロセスとサービスのモニタリングの推奨事項 - 372
プロセスにシグナルを送信する - 231
プロセスの nice 値を変更する (UNIX のみ) - 230
プロセス モニタ テーブル - 248, 317
プロセス モニタ テーブルのアクション パラメータ - 341
プロセス モニタ テーブルの表示 - 344
プロセス モニタ テーブルのフラグ - 333
プロセス モニタ テーブルの列 - 319
プロセス モニタの状態管理 - 314
プロセス モニタのトラップ - 69
プロセス モニタリング エントリの削除 - 370
プロセス モニタリングの設定 - 344
プロセス モニタリングの例 - 353
プロセス モニタリング用の edgewatch コマンド - 362
プロセッサ テーブル - 580
プロセッサ ロード トラッキング - 580
分散システム グループ - 243
別個の MIB 仕様で拡張変数に関する編集を行う方法 - 520
保守モード - 223
保守モードの設定 - 178
ホスト リソース システム グループ - 577
ホスト リソース ストレージ グループ - 578
ホスト リソース デバイス グループ - 578
ホスト リソースのインストール済みソフトウェア グループ - 584
ポリシーへの SystemEDGE モニタのインポート - 153

ま

マウントされたデバイスのマウント解除 - 229
ミラー テーブル - 250
メッセージバッファ グループ - 235
モニタおよび自動ウォッチャーの正規表現の例 - 192
モニタ対象オブジェクトの作成 - 549
モニタの例 - 89
モニタ プロセス - 548
モニタ保守ウィンドウ AIM - 532
モニタ保守ウィンドウ AIM の制限事項 - 533
モニタ保守ウィンドウ AIM の有効化 - 532
モニタ保守ウィンドウ モジュールの設定 - 191
モニタリング アーキテクチャ - 53
モニタリング機能 - 23
モニタリング設定 - 188

や

ユーザ テーブル - 230
用語 - 20
よくある問題と質問 - 554

ら

リモート コマンドの実行 - 232
リモート シェル グループ - 232
リモート設定および展開 - 91
履歴サンプリング - 493
履歴サンプリングの例 - 499
履歴制御テーブルと履歴データ テーブル - 494
履歴制御テーブルのエントリを管理するための **emphistory** コマンド - 504
履歴制御テーブルの設定 - 501
履歴制御テーブルの表示 - 501
履歴制御テーブルの列 - 495
履歴テーブル - 244
履歴テーブルの列 - 498
履歴の収集 - 75, 493
履歴の収集の概要 - 493
履歴の収集の仕組み - 76
履歴の収集の例 - 503
レガシー モードでのエージェントのインストール - 139
レガシー モードの操作 - 89
レガシー モニタ - 89
レジストリ データ - 523
連邦情報処理規格 (FIPS) モードの設定 - 175
ログ ファイルのモニタと、電子メールによる通知の送信 - 547
ログ ファイル モニタのトラップ - 73
ログ ファイル モニタリング - 72, 427
ログ ファイル モニタリングの概要 - 427
ログ ファイル モニタリングの推奨事項 - 454
ログ ファイル モニタリングの例 - 443
ログ ファイル モニタリング用の **edgwatch** コマンド - 450
ログ モニタ テーブル - 245, 429

ログ モニタ テーブルのアクション パラメータ - 438

ログ モニタ テーブルの表示 - 439
ログ モニタ テーブルのフラグ - 434
ログ モニタ テーブルの列 - 429
ログ モニタリング エントリの削除 - 453
ログ モニタリングの設定 - 440

わ

ワーク ロード マネージャ (WLM) クラスの
テーブル - 237