PANDORAFMS

仮想環境監視

仮想環境監視

Pandora FMS ドキュメント一覧に戻る

Amazon EC2®, DB2®, SAP® および VMware® を監視するには、Pandora FMS 自動検出 2.0 を利用します。

RHEV

Red Hat Enterprise Virtualization (RHEV) は、RedHat を利用するデータセンター企業で使われている仮想化技術の一つです。Pandora FMS は、RHEV Monitoring Plugin というプラグインを通してRHEV を使った仮想アーキテクチャをモニタすることができます。これは、簡単にRHEV の仮想アーキテクチャに関連するすべての値を制御することができます。

モニタする RHEV アーキテクチャ

このプラグインで、データセンター、ホストクラスタ、ストレージドメイン、ネットワーク、ホストおよび、仮想マシンといったRHEV アーキテクチャの全体をモニタすることができ、仮想環境の全体の状態を確認することができます。

これを実現するためにPandora では RHEV 仮想化システムによって公式に提供されている API を利用しています。

RHEV モニタリングプラグインでのモニタリング

RHEV 環境のモニタリングは、2つのコンポーネントに基づいています。

1. 自動検出動作を行うエージェントプラグインとデータ収集タスク。このエージェントプラグインは、Pandora FMS へ情報を送信します。
2. 検出したいくつかのパラメータを更新する自動検出スクリプト。このスクリプトは、「拡張」のために必要です。
3. RHEV View および RHEV Manager 拡張。これらは、プラグインに機能追加し、Pandora FMS のコンソールからインフラのモニタリングおよび仮想マシンの管理 (電源ON/OFF) ができるようにした「拡張」です。

自動検出スクリプトを利用するには、自動検出サーバを有効にする必要があります。

仮想マシンで収集するデータを API で返すようにするには、RHEVエージェントをインストールする必要があります。

詳細については RHEV のドキュメントを参照してください。

仮想マシンにインストールされた OS をモニタする には RHEV API ではなく Pandora FMS エージェント が必要です。

プラグイン動作の仕組

RHEV モニタリングプラグインは、RHEV 仮想環境の web API を通して情報を展開します。

モニタしたいだけであれば、この処理を実行するソフトウエアエージェントプラグインを設定します。

エージェントプラグインは、デバイスの自動検出を行い、検出したデバイスごとにモジュールを定義し XML を生成します。プラグインの設定では、どの要素をモニタリングしたいかを選択しモジュールを設定することができます。プラグインで作成されたモジュールの設定は変更が可能です。モジュールの、名前、説明、警告や障害状態の最小値、最大値を変更できます。

Pandora FMS 4.0 以降では XML を通して警告および障害状態の値(閾値)を更新することができますが、それより前のバージョンでは、ウェブコンソールから変更します。

XML が生成されると、エージェントプラグインは転送手法の設定に従って tentacle またはローカルでのファイルコピーにより、そのファイルを送信します。

また、RHEV View および RHEV Manager 拡張を利用する場合は、自動検出スクリプトを利用する必要があります。

自動検出スクリプトは、RHEV 仮想環境のそれぞれの Pandora FMS エージェントの値を更新します。これらの値は、RHEV View 拡張で状態を正しく表示したり RHEV Manager 拡張で仮想マシンを正しく管理するために必要な値です。

インストール要件

エージェントプラグインは、次のソフトウエアが必要です。

- curl
- perl-XML-Simple
- Pandora FMS ソフトウエアエージェント
- tentacle_client (ファイルの送信に tentacle を利用したい場合 tentacle_client は、Pandora FMS ソフトウエアエージェントと共に配布されています。)

Red Hat

RedHat システムでは、次のように依存ファイルをインストールします。

```
yum install perl-XML-Simple curl
```

SLES

SUSE システムでは、次のように依存ファイルをインストールします。

```
zypper install perl-XML-Simple curl
```

Debian/Ubuntu

Debian/Ubuntu システムでは、次のように依存ファイルをインストールします。

```
apt-get install libxml-simple-perl curl
```

RHEV 証明書のダウンロード

プラグインを実行する前に、RHEV API に HTTPS で接続するための証明書をダウンロードする必要があります。証明書をダウンロードするには、次のコマンドを実行します。

```
curl -o rhevm.cer http://[RHEVM-HOST]:8080/ca.crt
```

ここで、[rhevm-host] はRHEV API サーバのサーバ名です。例えば次の通りです。

```
curl -o rhevm.cer http://rhevm.server:8080/ca.crt
```

証明書をダウンロードしたら、次のコマンドで API に接続することができます。

```
curl -X GET -H "Accept: application/xml" -u [USER:PASS] --cacert [CERT]
https://[RHEVM-HOST]:8443/api
```

次の値を指定してください。

- USER: API へ接続する user@domain
- PASS: API へ接続するためのユーザのパスワード
- CERT: ダウンロードした証明書のパス
- RHEVM-HOST: API ホストのアドレス

実際の実行例は次の通りです。

```
curl -X GET -H "Accept: application/xml" -u [[user@testdomain:12345|]] --cacert
/home/user/ca.crt https://rhevm.server:8443/api
```

すべて成功すると、コマンドが XML フォーマットで　RHEV API の一般情報を出力します。

RHEV 設定時に考慮すべき点

RHEV の仮想化環境は、同じ名前で複数のエンティティを持つことができます。この機能では Pandora FMS において複数のエージェントが同じ名前でデータを送信してしまうという問題があります。この問題に加えて API が出力する XML のパースにおいて次のようなエラーが出る問題があります。

```
Warning: <data_center> element has non-unique value in 'name' key attribute:
Default at ./plugin-rhev.pl line 199
```

この問題の解決のためには RHEV 仮想環境のエンティティの名前付けポリシーを、同一名にならないようにする必要があります。

エージェントプラグインのインストール

エージェントプラグインをインストールするには rhev-plugin.pl および rhev-plugin.conf をプラグインを実行させたいマシンにインストールした pandora エージェントがアクセスできるフォルダにコピーするだけです。プラグインは、Pandora FMS サーバと同じマシンまたは他のマシンにインストールされたエージェントより実行できます。

プラグインを実行するには、エージェントの設定ファイル(デフォルトでは /etc/pandora/pandora_agent.conf です)に次のような行を追加します。

```
module_plugin /root/rhev-plugin.pl /root/rhev-plugin.conf
```

この行を追加することにより、エージェントの実行間隔でエージェントプラグインが実行されます。

RHEV 仮想アーキテクチャのモニタリング

プラグインの実行結果を見るには、操作(Operation) → モニタリング(Monitoring) → 表示(Views) → エージェント詳細(Agent Detail) をクリックします。エージェントの名前をクリックすると、プラグインによって作成された監視モジュールやその他の関連データを確認できます。

このプラグインは、RHEV アーキテクチャの検出で検出されたエンティティごとに Pandora FMS にエージェントを作成します。 エンティティのタイプごとに、一連の特定のモジュールが自動的に作

成され、それぞれの重要な情報が監視されます。

選択したエージェントが仮想マシンではなくホストに対応する場合、監視モジュールは異なります。

RHEV プラグインは、仮想アーキテクチャ内で発生するイベントも監視します。 プラグインは、影響を受ける各エンティティ内の監視対象イベントごとにモジュールを作成します。 イベントから作成されたモジュールのデータは、イベントデータ (イベント時間、イベントの説明) です。

RHEVアーキテクチャ自体に関連するエージェントとモジュールに加えて、プラグインを実行するエージェント内に、デフォルトで RHEV Plugin と呼ばれるモジュールが生成されます。

エンティティ状態のモニタリング

エンティティの状態モジュールは、RHEVアーキテクチャで事前に定義された値を返します。これは、モニタするエンティティの状態に応じて、値が up, down, error, maintenance, non_operational などの文字列であることを意味します。

警告や障害状態を定義するには、モジュール設定にて正規表現を利用する必要があります。例えば、値が *error* *down* *non_operational* の時に障害状態とするには、そのモジュールの障害の場合の文字列(Str.) フィールドに、次の正規表現を設定します。

```
error|down|non_operational
```

Pandora FMS 4.0 の初期バージョンではこのオプションは利用できませんが、同一の場合の条件を利用して設定することができます。上記の例でアラーとテンプレートを作成するには、次の手順を実施します。

1. 優先度が障害のアラートテンプレートを作成し、条件種別(Condition Type) フィールドを 正規表現(Regular expresion) に設定します。
2. error|down|non_operational という正規表現をフィールドに入力します。これは、モジュールの値が *error* *down* *non_operational* のいずれかになった場合にアラートを実行するという意味になります。
3. 以降は通常の設定を行います。

テンプレートを定義すると、イベント作成、メールや SMS 送信など、アラートが発生したときの実行アクションを選択することができます。

RHEV アーキテクチャのためのエージェントモジュール

以下に、RHEV アーキテクチャのそれぞれの要素のためのモジュールの詳細について示します。

データセンター

- *Status*: データセンターの状態

ストレージドメイン

- *Available Space*: ストレージドメインの空き容量
- *Committed Space*: ストレージドメインのコミット容量
- *Used Space*: ストレージドメインの利用容量
- *Percent Free Space*: ストレージドメインの空き容量率

ネットワーク

- *Status*: 仮想ネットワークの状態
- *STP Status*: スパニングツリープロトコルの状態

クラスタ

- *Overcommit Percent*: クラスタのオーバーコミット率
- *Transparent HugePages*: Transparent HugePage の状態
- *High threshold*: ポリシープランニングのための上位閾値
- *Low threshold*: ポリシープランニングのための下位閾値
- *Threshold duration*: ポリシープランニングのための閾値期間

ホスト

- *Status*: ホストの状態
- *Buffers size*: バッファサイズ
- *Cache size*: キャッシュサイズ
- *Cached swap*: キャッシュスワップのためのメモリ量 (バイト単位)
- *Free memory*: 空きメモリ量 (バイト単位)
- *Percent free memory*: 空きメモリ率
- *Swap cached percent*: キャッシュスワップメモリ率
- *Swap free*: スワップの空き容量 (バイト単位)
- *Swap free percent*: 空きスワップメモリ率
- *Total Memory*: このホストのトータルメモリ容量 (バイト単位)
- *Total Swap*: スワップメモリ容量 (バイト単位)
- *Used memory*: 利用メモリ量 (バイト単位)
- *Used Swap*: 利用スワップメモリ量 (バイト単位)
- *Nic [x] TX*: nic x の送信速度 (バイト/秒) インタフェースごとにモジュールが生成されます。
- *Nic [x] RX*: nic x の受信速度 (バイト/秒) インタフェースごとにモジュールが生成されます。
- *Nic [x] erros TX*: nic x の送信エラー数。インタフェースごとにモジュールが生成されます。
- *Nic [x] erros RX*: nic x の受信エラー数。インタフェースごとにモジュールが生成されます。
- *User CPU*: user CPU 使用率
- *System CPU*: system CPU 使用率
- *CPU Idle*: idle CPU 使用率
- *CPU Load*: 5分間のロードアベレージ
- *KSM CPU*: KSM の CPU 使用率
- *Active VM*: ホスト内の稼働中の仮想マシン数
- *Migrating VM*: ホスト内でマイグレーション処理中の仮想マシン数
- *Total VM*: このホストにおける全仮想マシン数
- *Fence Status*: ホストフェンスの状態

仮想マシン

- *Status*: 仮想マシンの状態
- *Disk [x] read*: ディスク x の読み込み速度 (バイト/秒)。ディスクごとにモジュールが生成されます。
- *Disk [x] write*: ディスク x の書き込み速度 (バイト/秒)。ディスクごとにモジュールが生成されます。
- *Disk [x] size*: disk x のディスクサイズ。ディスクごとにモジュールが生成されます。
- *Disk [x] status*: disk x の状態。ディスクごとにモジュールが生成されます。
- *Nic [x] TX*: nic x の送信速度 (バイト/秒)nic ごとにモジュールが生成されます。
- *Nic [x] RX*: nic x の受信速度 (バイト/秒)nic ごとにモジュールが生成されます。
- *Nic [x] erros TX*: nic x の送信エラー数。nic ごとにモジュールが生成されます。
- *Nic [x] erros RX*: nic x の受信エラー数。nic ごとにモジュールが生成されます。
- *Installed memory*: 設定されたメモリ容量 (バイト単位)
- *Percent free memory*: 空きメモリ率
- *Used memory*: 利用メモリ量 (バイト単位)
- *Stateless*: ステートレス機能の状態
- *HA Status*: HA 機能の状態
- *Total CPU*: この仮想マシン全体の CPU 使用率
- *Hypervisor CPU*: 仮想マシンによって使われている、ハイパーバイザーの CPU 使用率
- *Guest CPU*: 仮想マシンの CPU 使用率

イベント

- *Event [x]*: システム内で発生したイベント x の説明。それぞれのエージェント内に、検出されたイベントごとにモジュールが作成されます。

RHEV アーキテクチャの管理と参照

この節では、インストール、設定、および、どのように RHEV View および RHEV Manager 拡張が動作するかを説明します。

RHEV View および RHEV Manager 拡張は、Pandora FMS 4.0.2 以上 でのみ動作します。

自動検出タスクのインストール

カスタム検出タスクの作成は、検出サーバ を参照してください。

RHEV View および RHEV Manager 拡張のインストール

拡張のインストールは、extensions フォルダの中身をコピーするだけです。プラグインを展開したものを Pandora FMS コンソールの extensions フォルダへコピーします。実行コマンドは次の通りです。

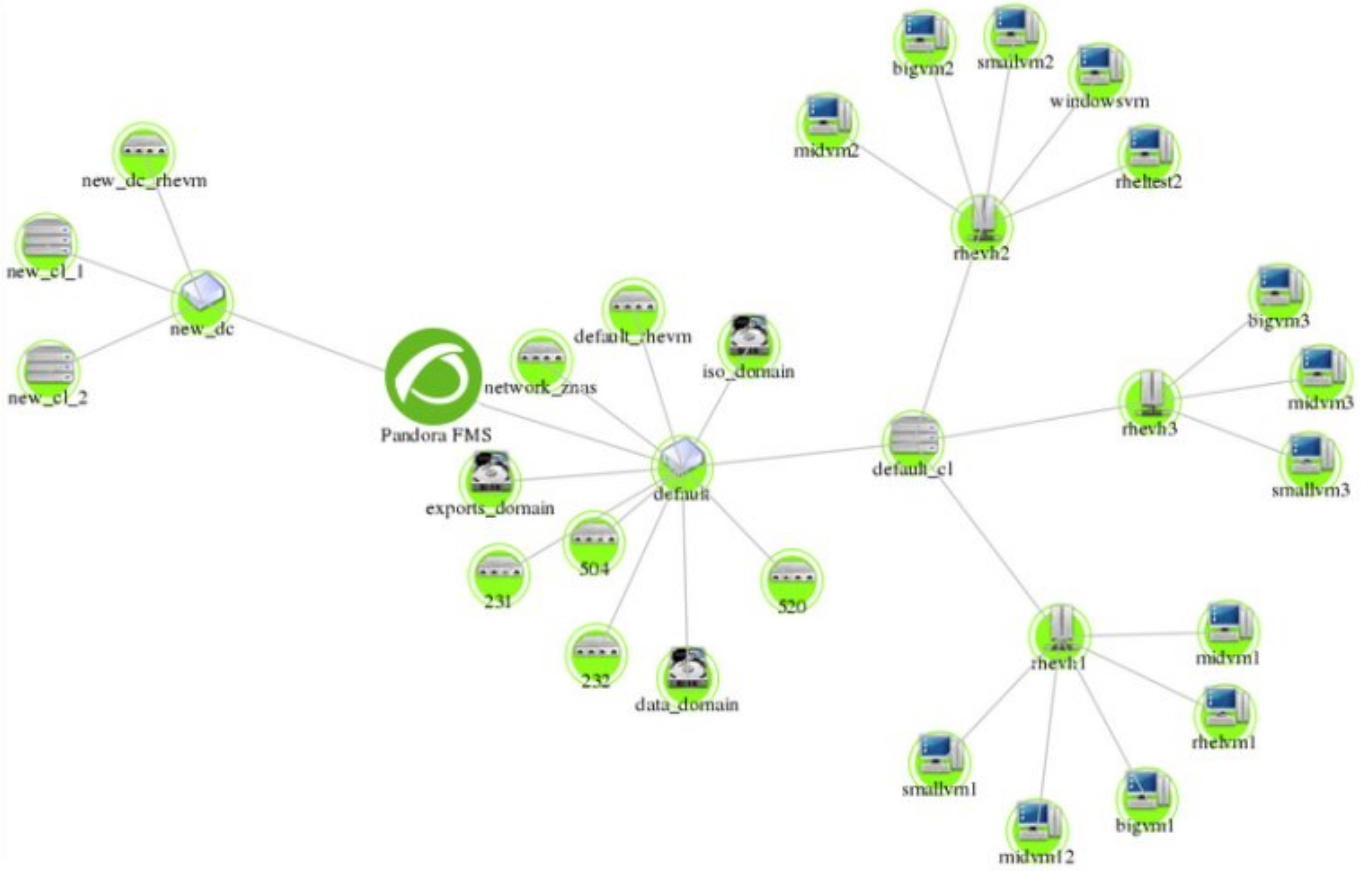
```
cp -R extensions/* <pandora_console_dir>/enterprise/extensions/
```

これでRHEV モニタリング拡張が使えるようになります。

RHEV View 拡張の利用

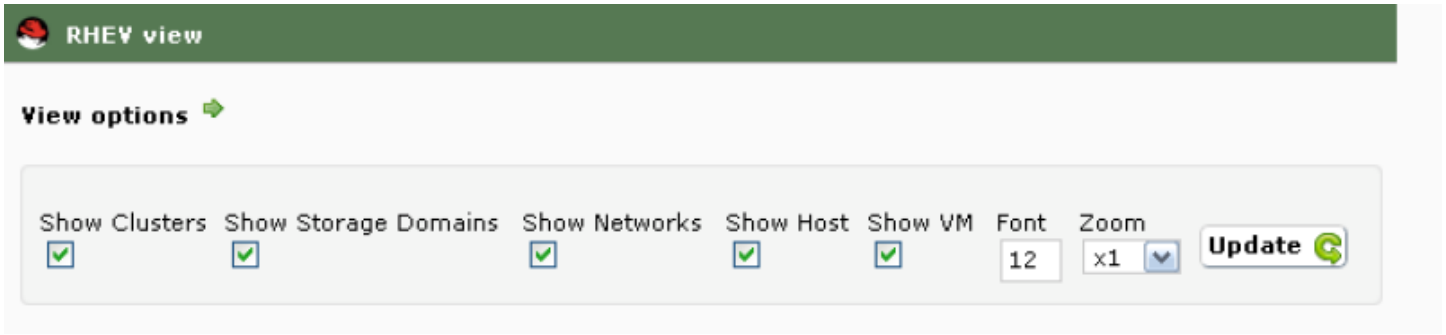
RHEV View 拡張を利用するには、*モニタリング(Monitoring)* メニューの中の *RHEV View* をクリックします。

拡張は、プラグインで検出した RHEV アーキテクチャの全コンポーネントを次のようにマップ表示します。



マップ内には□RHEVアーキテクチャの異なる要素(データセンター、ストレージドメイン、クラスタ、ネットワーク、ホスト、仮想マシン)が表示されます。それぞれの要素はアイコン表示され、それぞれの要素ごとに異なります。アイコン間の関係は、RHEV アーキテクチャの要素間の関係として表示します。この画面では、一目で要素の関係と状態を見ることができます。

拡張には、表示を設定するためのメニューがあります。エンティティを隠したり表示したり、テキストのサイズを変更したり、詳細画像の拡大・縮小ができます。



以下の画像の例では、ネットワーク、ホスト、仮想マシンのエレメントは隠れています。なぜなら、データセンターで、クラスタとストレージドメインの関係の詳細を見たいからです。

RHEV Manager 拡張の利用

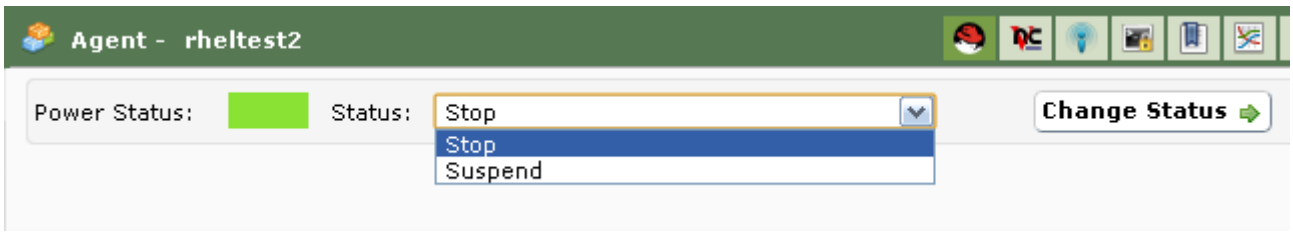
RHEV Manager 拡張は、Pandora FMS 内で RHEV 仮想マシンを表示するエージェント操作画面にあります。

この拡張は、curl コマンドを利用します。このコマンドがインストールされ、Pandora FMS コンソールを実行する web サーバプロセスのユーザ権限で利用できる必要があります。

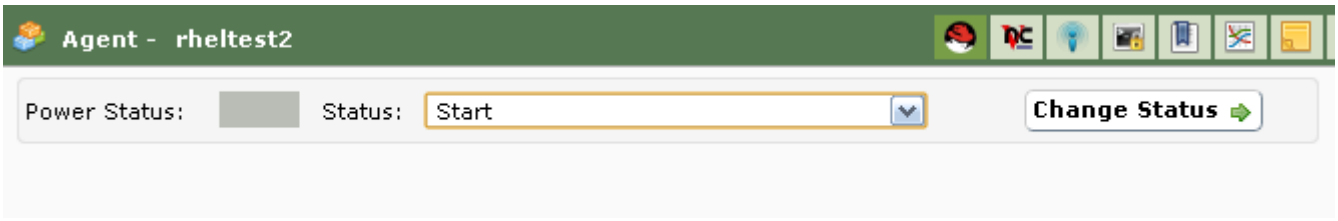
拡張にアクセスするには、エージェントのタブにある Red Hat ロゴのアイコンをクリックします。

拡張では、RHEV 管理コンソールを使わずに仮想マシンを管理(電源 on/off、サスペンド)することができます。拡張は、色で仮想マシンの現在の状態を表示(緑 = 電源on、オレンジ = サスペンド、グレー

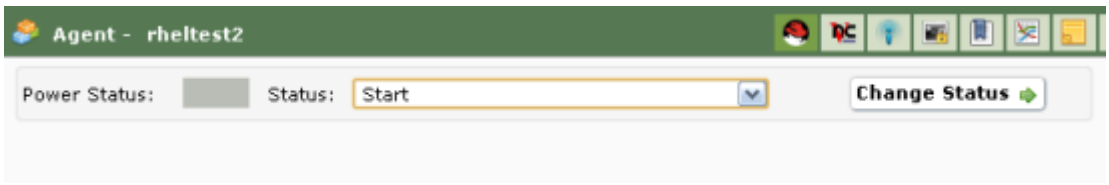
= 電源off)します。また、状態を選択し、状態変更(Change Status) ボタンをクリックして状態を変更することができます。

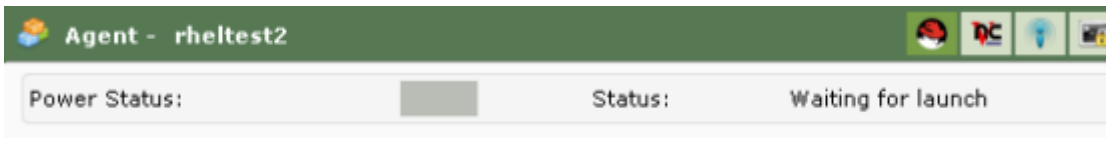


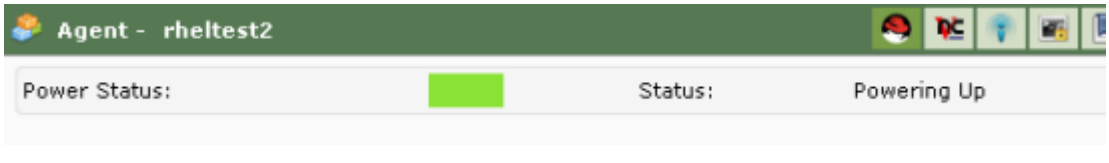
仮想マシンを停止するために停止(Stop) 状態を選択すると、拡張は RHEV API に接続しコマンドを送信します。結果、仮想マシンの状態が変化し、選択オプションが次のようになります。

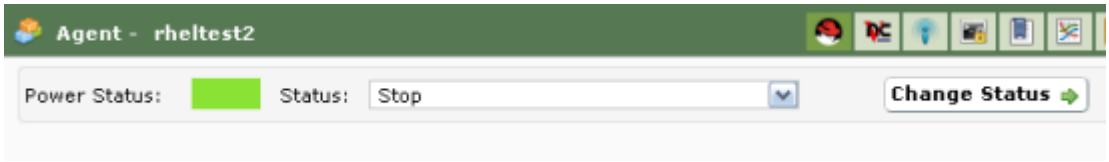


いくつかの状態の間の変更には、いくつかのステップがあります。例えば、停止(Stop) から 開始(Start) への変更です。この場合、拡張はそれぞれのステップで仮想マシンの状態を表示します。停止から開始への変更では、仮想マシンの状態は次のように変化します。









エージェントプラグイン設定

エージェントプラグイン設定は、デフォルトで rhev-plugin.conf という設定ファイル使って行います。

デフォルトでは、エージェントプラグインは全エンティティを選択し、名前と説明をデフォルトの値で全モジュールを作成します。これらのパラメータはすべて設定ファイルを通してカスタマイズできます。

設定ファイル

設定ファイルは、全体の設定と個々の監視の設定の 2つの部分から成ります。

全体の設定の部分では、Configuration という記述から始まり、プラグイン設定に関する情報があります。この部分で設定可能なパラメータは次の通りです。

- *module_name*: プラグインを実行するエージェントから報告されるモジュール名です。
- *server*: RHEV API を実行するホスト名です。
- *user*: API に接続するユーザです user@domain という書式です。
- *pass*: API へ接続するパスワードです。
- *cert*: API 証明書のパスです。
- *temporal*: テンポラリフォルダです。
- *logfile*: ログファイル名です。
- *transfer_mode*: 転送モードです local または tentacle です。
- *tentacle_ip*: 情報の送信先の tentacle サーバの IP です。一般的に Pandora サーバと同じマシンです。転送モードに tentacle を利用した場合に利用できます。
- *tentacle_port*: tentacle サーバのポート番号です。転送モードに tentacle を利用した場合に利用できます。
- *tentacle_opts*: tentacle サーバの拡張オプションです。転送モードに tentacle を利用した場合に利用できます。

監視設定部分には、いくつかのサブセクションがあります。一つ目は、Reject というトークンがあり、除外する仮想環境のエンティティの名前のリストを作成することができます。エンティティを除外するには、次のように名前のリストを記載する必要があります。

```
#Dismissed entities
Reject
mv1
mv_WindowsXP
mv_WebServer1
...
```

全ホスト、全仮想マシンなど、一つの種類のすべてのエンティティを除外することもできます。それぞれのエンティティのトークンは、all_dc (データセンタ) all_host (ホスト) all_network (ネットワーク) all_storage (ストレージドメイン) all_cluster (クラスタ) all_vm (仮想マシン)です。これらのトークンの利用例を以下に示します。

```
#Dismissed entities
Reject
```

```
all_dc
all_host
all_network
all_storage
all_cluster
all_vm
```

2つ目のセクションは、Rename というトークンで定義され、エンティティの名前を変更できます。この機能は、Pandora FMS の同一エージェント内でソフトウエアエージェントと API 情報を結びつけるのにとても便利です。このセクションの設定は、古い名前、新しい名前の順にスペースを入れて次のように記述します。

```
#Rename entities
Rename
mv_WebServer1 WebServer1
mv_WindowsXP WindowsXP Test
...
```

次のサブセクションは、エンティティのモニタリング設定に関するものです。それぞれのエンティティには、それぞれ DataCenter, StorageDomain, Network, Cluster, Host および VM というトークンがあります。各エンティティで、モジュールの有効・無効、警告や障害状態となる最小値や最大値を定義することができます。例を以下に示します。

```
#VM Modules
VM
status disabled
errors_total_tx name = TX Error Net [%s]; desc = Total error TX net; limits =
60 70 71 100
memory_used name = Used Mem; desc = Memory used by the virtual machine; limits
= 256 1024 1025 2048
...
```

それぞれの行はモニタリングモジュールに関連していて、次の 2つのオプションがあります。

- <モジュール> disabled: モジュールは、作成されません
- <モジュール> name = <名前>; desc = <説明>; limits = <警告の最小値> <警告の最大値> <障害の最小値> <障害の最大値 » モジュールは、指定した名前と説明で作成されます。また、警告および障害の最大・最小の閾値を指定します。

設定ファイルの行構造 に注意し、特にモジュール名とモジュールの説明の近くの; 文字 に注意することがとても重要 です。これらの行は、同じではありません。(; 文字の前のスペースを確認してください)

```
errors_total_tx name = TX Error Net [%s]; desc = Total error TX net; limits =
60 70 71 100 #OK
errors_total_tx name = TX Error Net [%s]    ; desc = Total error TX net    ;
limits = 60 70 71 100 #Wrong
```

モジュールは、コマンドラインで入力しやすい短い名前で参照されます。フルネームと短い名前の

関連付けした表を次の章に示します。

仮想マシンの設定例 (VM セクション) を見てみましょう。

仮想マシンをモニタするには、設定ファイルの VM セクションで、モジュールを有効化・無効化するかの一覧を定義します。*status* モジュールは無効化されており、*errors_total_tx* および *memory_used* モジュールはカスタム値を持っています。リストに無い残りのモジュールは、デフォルトの値で作成されます。この設定で、memory_used モジュールは次の値になります。

- *Name*: Used Memory
- *Description*: Memory used by the virtual machine
- *Min Warning*: 256
- *Max Warning*: 1024
- *Min Critical*: 1025
- *Max Critical*: 2048

ディスクやネットワークインタフェースに関連するモジュールは動的に生成されます。それぞれの要素ごとにモジュールが作成され、次のようにモジュール名は特別な書式になっています。

```
errors_total_tx name = Errores TX Net [%s]; desc = Errores totales TX de red;
limits = 60 70 71 100
```

この場合、名前は動的に決まる部分があり、% マクロを利用することができます。これはプラグインによりモジュール名で動的に変わるものに置き換えられます。

例えばerrors_total_tx モジュールのデフォルトの名前は次の通りです。

```
Nic [nic1] errors TX
```

この設定での名前は次の通りです。

```
TX Error Net [nic1]
```

nic1 がモジュール名で動的に決まる部分です。

設定ファイルに関連する全てのエラーはログファイルに出力されます。またPandora FMS のプラグインを実行するエージェント内部の非同期モジュールへ送信されます。

それぞれの要素に関連したセクションに加え、設定ファイルにはイベント設定のための共通セクションがあります。このセクションは、EventCodes というトークンで定義され、モニタするすべてのイベントコード一覧を次のように定義します。

```
EventCodes
30
920
```

```
980
509
956
```

このセクションを定義しないと、イベントモニタリングは動作しません。

複数のソフトウエアエージェントでのモニタリング負荷の分散

設定ファイルを通して RHEV 仮想環境のモニタリング負荷を分散することができます。

そのためには、モニタするエンティティをエージェント間で分割する必要があります。この例では、次のようなアーキテクチャを想定します。

```
DC1
 |
 |- Cluster 1.1
       |- c1.1mv1
       |- c1.1mv2
       |- c1.1mv3

 |- Cluster 1.2
       |- c1.2mv1
       |- c1.2mv2
       |- c1.2mv3

DC2
 |
 |- Cluster 2.1
       |- c2.1mv1
       |- c2.1mv2
       |- c2.1mv3

 |- Cluster 2.2
       |- c2.2mv1
       |- c2.2mv2
       |- c2.2mv3
```

負荷を分散する方法として、一つのデータセンタをそれぞれ違うエージェントに割り当てます。そのためには、エンティティを除外する機能 (Reject トークン) を利用します。

最初のエージェントはデータセンタ DC1 のみをモニタし、DC2 のエンティティは除外します。

```
Reject
DC2
Cluster 2.1
Cluster 2.2
c2.1mv1
c2.1mv2
```

```
c2.1mv3
c2.2mv1
c2.2mv2
c2.2mv3
```

2つ目のソフトウエアエージェントは、データセンタ DC2 をモニタし、DC1 は除外します。

```
Reject
DC1
Cluster 1.1
Cluster 1.2
c1.1mv1
c1.1mv2
c1.1mv3
c1.2mv1
c1.2mv2
c1.2mv3
```

また、クラスタをベースに負荷を分散することもできます。例えば、4つのソフトウエアエージェントがあり、それぞれ異なるクラスタをモニタします。

ソフトウエアエージェント1 ではクラスタ 1.1 をモニタし、他のエンティティを除外します。

```
Reject
DC1
Cluster 1.2
c1.2mv1
c1.2mv2
c1.2mv3
DC2
Cluster 2.1
Cluster 2.2
c2.1mv1
c2.1mv2
c2.1mv3
c2.2mv1
c2.2mv2
c2.2mv3
```

ソフトウエアエージェント2 ではクラスタ 1.2 をモニタし、他のエンティティを除外します。

```
Reject
DC1
Cluster 1.1
c1.1mv1
c1.1mv2
c1.1mv3
DC2
Cluster 2.1
Cluster 2.2
```

```
c2.1mv1
c2.1mv2
c2.1mv3
c2.2mv1
c2.2mv2
c2.2mv3
```

ソフトウエアエージェント3ではクラスタ 2.1 をモニタし、他のエンティティを除外します。

```
Reject
DC1
Cluster 1.1
Cluster 1.2
c1.1mv1
c1.1mv2
c1.1mv3
c1.2mv1
c1.2mv2
c1.2mv3
DC2
Cluster 2.2
c2.2mv1
c2.2mv2
c2.2mv3
```

ソフトウエアエージェント4ではクラスタ 2.2 をモニタし、他のエンティティを除外します。

```
Reject
DC1
Cluster 1.1
Cluster 1.2
c1.1mv1
c1.1mv2
c1.1mv3
c1.2mv1
c1.2mv2
c1.2mv3
DC2
Cluster 2.1
c2.1mv1
c2.1mv2
c2.1mv3
```

エンティティの除外設定はとても柔軟で、それぞれのソフトウエアエージェントで複数のエンティティのモニタリング負荷を分散させることができます。

設定ファイル例

全モジュール無効化の例

```
#These lines are comments

#Plugin configuration parameters
Configuration
server rhevm.server
user user@testdomain
pass 12345
cert /home/user/rhevm.cer
temporal /tmp
logfile /tmp/plugin-rhev.log
transfer_mode local
tentacle_ip 127.0.0.1
tentacle_port 41121
tentacle_opts

#Dismissed entities
Reject

#Data Center modules
DataCenter
status disabled

#StorageDomain modules
StorageDomain
available disabled
used disabled
committed disabled
free_percent disabled

#Networks modules
Network
status disabled
stp disabled

#Clusters modules
Cluster
overcommit disabled
hugepages disabled
threshold_low disabled
threshold_high disabled
threshold_duration disabled

#Host Modules
Host
status disabled
vm_active disabled
vm_migrating disabled
vm_total disabled
data_current_rx disabled
data_current_tx disabled
errors_total_rx disabled
errors_total_tx disabled
```

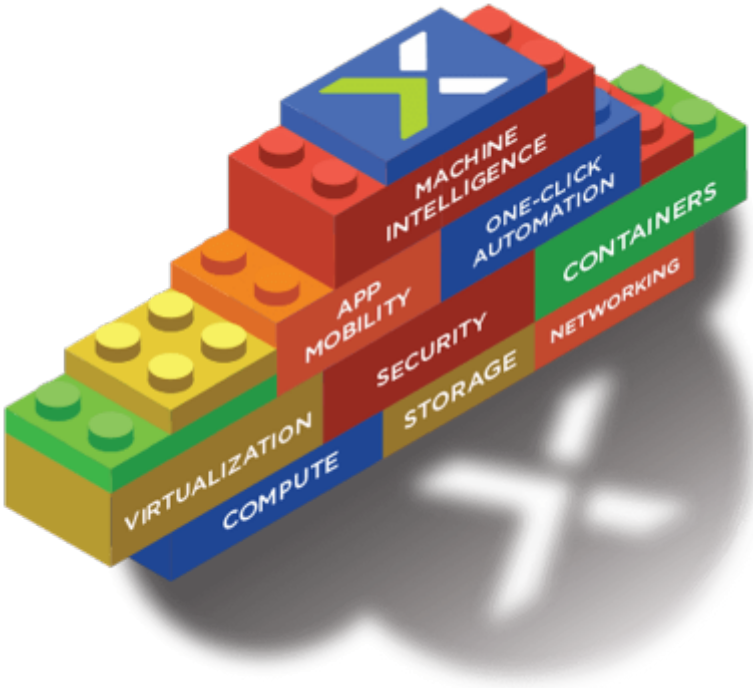
```
memory_cached disabled
memory_total disabled
swap_free_percent disabled
swap_cached_percent disabled
swap_free disabled
cpu_current_idle disabled
cpu_current_user disabled
memory_used disabled
ksm_cpu_current disabled
memory_free_percent disabled
swap_total disabled
memory_buffers disabled
cpu_current_system disabled
cpu_load_avg_5m disabled
swap_cached disabled
swap_used disabled
memory_free disabled
fence_status disabled

#VM Modules
VM
status disabled
stateless disabled
ha disabled
cpu_current_guest disabled
cpu_current_hypervisor disabled
memory_free_percent disabled
memory_installed disabled
memory_used disabled
cpu_current_total disabled
data_current_read disabled
data_current_write disabled
size disabled
disk_status disabled
data_current_rx disabled
data_current_tx disabled
errors_total_rx disabled
errors_total_tx disabled
```

Nutanix

Nutanix ハイパーコンバージドソリューションでは、ネットワーク、ディスク、処理、メモリリソースのすべてを一ヵ所で管理することができます。

Pandora FMS の Nutanix 監視プラグインにより□Nutaix ソリューションの状態を常に監視することができます。



プラグイン操作

Nutanix プラグインは、Perl で書かれたプログラムで、Nutanix PRISM の REST API へ接続します。以下の要素を監視するために必要なメトリックスを取得します。

- Nutanix クラスタ
- ストレージデバイス
- コンテナ
- 仮想マシン
- ホスト
- レプリケーションプロセスの状態

プラグインの必要条件

REST API から情報を取得するためには、以下が必要です。

- ポータルの IPアドレス/FQDN
- API の読み出し権限を持った ユーザ
- ユーザの *パスワード*

監視結果を Pandora FMS へ反映させるために、以下が必要です。

- ローカルまたは Tentacle での *情報の転送*
  - ローカルであれば、結果を含む XML ファイルを置くディレクトリが必要です。またディレクトリには書き込み権限が必要です。
  - Tentacle を利用する場合は、Pandora サーバの IPアドレスまたは FQDN に Tentacle のポートで接続できる必要があります。tentacle クライアントの場所やオプションの定義も必要です。

プラグインのインストール

プラグインに必要なファイルを モジュールライブラリ からダウンロードします。

Nutanix インフラストラクチャを監視するリモートのコンピュータへファイルを転送し、プラグインファイルを展開します。

```
tar xvzf pandora_nutanix.tar.gz
```

プラグイン設定

次のフィールドがあります。

Nutanix API 設定

- nx_fqdn: メインの Prism サーバのアドレス。
- nx_port: REST API が公開されるポート (デフォルトは 9440)。
- nx_user: REST API を介した読み取り権限を持つユーザ。
- nx_pass: 当該ユーザのパスワード。
- use_https: https を使用する (1) かしないか (0)
- nx_rest_version: REST API バージョン (デフォルトは 'v1')

Nutanix エージェント設定

- agent_interval: プラグインによって生成されるエージェントの間隔 (デフォルトは 300)。
- agent_group: 生成されたエージェントが属するグループ (PandoraServer 設定で「autocreate_group」がコメントアウトされている場合)、デフォルトは Nutanix
- module_interval: 生成されたエージェントのモジュールの間隔 (倍率、デフォルトは 1)。
- module_tags: 生成されたエージェントの新しいモジュールに関連付けられるタグ。
- module_group: 新しいモジュールが属するグループ。

Pandora FMS サーバとの通信設定

- mode: データ転送モード“local” または “tentacle”
- tentacle_ip: Pandora FMS サーバの IP アドレスTentacle モードでのみ有効。
- tentacle_port: Tentacle サービスが待ち受けているポート。
- tentacle_opts: Tentacle サービスの追加設定オプション。
- tentacle_client: tentacle クライアントのフルパス。
- temp: 一時作業ディレクトリ。
- local_folder: “local” データ転送モードの際のファイル転送パス。

フィルタ

- cluster_monitoring: クラスタ監視の有効化(1)、無効化(0)。
- storage_monitoring: ストレージデバイス監視の有効化(1)、無効化(0)。
- container_monitoring: ストレージコンテナ監視の有効化(1)、無効化(0)。
- vm_monitoring: 仮想マシン監視の有効化(1)、無効化(0)。
- host_monitoring: 仮想サーバ (Nutanix ノード) 監視の有効化(1)、無効化(0)。

- pd_monitoring: 保護ドメイン監視の有効化(1)、無効化(0)。

カスタマイズ

- cluster_agent_header: クラスタデバイスエージェントの名前のヘッダ。
- storage_agent_header: ストレージデバイスタイプのデバイスエージェントの名前のヘッダ。
- host_agent_header: 仮想マシンサーバタイプのデバイスエージェント (Nutanix ノード) の名前のヘッダ。
- container_agent_header: ストレージコンテナタイプのデバイスエージェントの名前のヘッダ。
- vm_agent_header: 仮想マシンタイプデバイスエージェントの名前のヘッダ。
- pd_agent_header: 保護ドメインタイプデバイスエージェントの名前のヘッダ。

モジュール生成ルール

- vm_stat: 仮想マシンを監視するためのモジュールを追加するためのルール。デフォルトでは `hypervisor_cpu_usage_ppm|hypervisor_memory_usage_ppm|.*avg.*`。これは、メトリック名がこのフィールドに示された正規表現と一致する場合に生成される特別なモジュールを示します。値 `.*` を設定すると、利用可能なすべてのメトリックを監視します。
- host_stat: 仮想マシンサーバ (Nutanix ノード) を監視するためのモジュールを追加するためのルール。デフォルトでは `hypervisor_cpu_usage_ppm|hypervisor_memory_usage_ppm|.*avg.*`。これは、メトリクス名が次の正規表現に一致する場合に生成される特別なモジュールを示します。 値 `.*` を設定すると、利用可能なすべてのメトリックを監視します。
- pd_stat: 保護ドメインを監視するためのモジュールを追加するためのルール。デフォルトでは `replication_transmitted_bandwidth_kBps|replication_total_transmitted_bytes` です。これは、メトリック名がこのフィールドに示されている正規表現と一致する場合に生成される特別なモジュールを示します。 値 `.*` を設定すると、利用可能なすべてのメトリックを監視します。

エンティティの名前変更

エンティティの名前変更

- RENAME aaa TO bbb: エンティティの名前変更のルール。名前を変更する必要がある要素と同じ数のディレクティブを定義できます。

エンティティの実行

- REJECT aaa: エンティティ監視を除外するためのルール。除外する必要がある要素と同じ数のポリシーを定義できます。

プラグインの実行

プラグインを実行するサーバは、監視のために Pandora FMS サーバおよび Nutanix® インフラストラクチャの両方にアクセスできる必要があります。

手動実行:

```
./pandora_nutanix-linux-x64 pandora_nutanix.conf
```

`/etc/crontab` に以下の設定を追加することにより cron でプラグインを自動実行できます。

```
/5 * * * * root /path/to/plugin/pandora_nutanix-linux-x64
/path/to/plugin/pandora_nutanix.conf
```

XenServer

XenServer プラグインの動作

Xen 環境監視のための Pandora FMS プラグインは、Python で書かれています。 必要な情報をすべて取得するにはXenAPI を使用します。 これにより、次のタイプの要素の監視が可能になります。

- Xen 内の仮想化システム
- ストレージリソース
- XenServer 6.5 および 7.2 自身(ホスト)

プラグインに必要なもの

プラグインを実行するシステムには次のものが必要です。

- Python のインストール
- 次の Python ライブラリのインストール: XenAPI, xmltodict
- XenServer API へのアクセス (プラグインを実行するマシンからXenServer へのポート 443 または 80 での接続許可)
- 利用できる情報が多いため、仮想マシンには Xen Server Tools のインストールを推奨します。

プラグインのインストール

Pandora FMS の XenServer プラグインを、モジュールライブラリ からダウンロードします。

実行環境のマシン(Windows または Linux)で、Pandora FMS エージェントまたはシステム cron を使用して実行できるディレクトリにファイルを展開します。

プラグイン設定

Xen 用の Pandora FMS プラグイン設定は以下の通りです。

設定ブロック [CONF]

- xen_server_ip: Xen Server の IP/FQDN アドレス
- user: Xen API に対してクエリを出せるユーザ
- password: ユーザのパスワード
- temporary: テンポラリディレクトリ

設定ブロック [PANDORA]

- tentacle_client: Tentacle クライアントの実行ファイルの場所

- tentacle_ip: Tentacle の接続先 IP アドレス
- tentacle_port: Tentacle の接続先ポート
- logfile: ログファイルのパス
- interval: 生成したエージェントの間隔
- group: 生成したエージェントに割り当てるグループ

設定ブロック [TUNNING]

- time_adjustment: プラグインを実行しているコンピュータと Xen server との間で許容可能な時間差を調整するパラメータ。(デフォルト = 10, 秒単位)
- scan_vm_ip: プラグインが Xen サーバーの VM の IP を取得するかどうかを定義するパラメータ□XenTools がインストールされた VM の IP のみ取得できます。有効化(scan_vm_ip = true)または、無効化(scan_vm_ip = false)の設定ができます。設定されていない場合は、有効になります。

設定ブロック [RENAME]

- xen_element_name=pandora_agent_name:このブロックでは、以下のフォーマットで多くのエントリーを定義できます□XenServer 要素の名前を Pandora で使われるエージェント名に変更することができます□VMs□SR□Xen Server 自身を変更することができ、以下に例を示します。

```
[RENAME]
example-xen-server = Example Xen Server
Example Xen Server 2 = example-xen-server-2
example-vm = Example VM
Example VM 2 = example-vm-2
example-sr = Example SR
Example SR 2 = example-sr-2
```

名前にスペースが含まれていても、クオートでくくる必要はありません。

プラグインの実行

Pandora FMS エージェントの設定に以下を追加することによって、プラグインの実行ができます。

```
module_plugin python "<path>\xen-plugin.py" "<path>\xen-plugin.conf"
```

システムの cron で設定する場合は、/etc/crontab に以下を追加します。

```
/5 * * * * root python "<path>\xen-plugin.py" "<path>\xen-plugin.conf">
/dev/null 2>&1
```

プラグインを手動実行すると、出力は以下のようになります。

```
python "<path>\xen-plugin.py" "<path>\xen-plugin.conf"
<module>
<name><![CDATA[XenServer Plugin]]></name>
<type><![CDATA[async_string]]></type>
<description><![CDATA[Result of XenServer Plugin execution]]></description>
```

```
<data><![CDATA[OK]]></data>
</module>
```

OpenNebula

OpenNebula プラグインの動作

OpenNebula 環境を監視するための Pandora FMS プラグインは Perl で書かれています。OpenNebula サーバのローカルで実行し、OpenNebula 自身の管理コマンドを使って必要な情報を取得します。

- クラスタ
- ホスト
- 仮想マシン
- ストレージリソース

プラグインに必要なもの

プラグインの実行には以下が必要です。

- Perl のインストール
- 以下のコマンドの実行権限
    - onehost
    - onecluster
    - onedatastore

プラグインの実行は OpenNebula システム 5.X.X でテスト
しています。

プラグインのインストール

OpenNebula 用の Pandora FMS プラグインは、モジュールライブラリ からダウンロードします。

Pandora FMS エージェントまたはシステム cron を使用して実行できるディレクトリにファイルを展開します。

```
unzip pandora_OpenNebula.zip
```

プラグインの設定

Pandora サーバとの通信設定

- mode: データ転送モード。“local” または “tentacle” です。

- tentacle_ip: Pandora サーバの IP アドレス（tentacle モードの場合のみ）。
- tentacle_port: Tentacle サーバのポート番号
- tentacle_opts: Tentacle の追加オプション
- tentacle_client: Tentacle クライアントのパス
- temp: テンポラリディレクトリ
- local_folder: データ転送モードが “local” の場合のパス

エージェント設定

- agent_interval: エージェント間隔。デフォルトは 300。
- agent_group: エージェントグループ。デフォルトは OpenNebula（

モジュールカスタマイズ

- module_group: モジュールグループ。デフォルトは、OpenNebula（
- module_interval: モジュール間隔(倍率)。デフォルトは 1
- module_tags: モジュールのタグ

名前のカスタマイズ

- cluster_agent_header: クラスタタイプデバイスエージェントのエージェント名ヘッダ
- host_agent_header: 仮想マシンサーバタイプデバイスエージェントのエージェント名ヘッダ
- storage_agent_header: ストレージデバイスタイプエージェントのエージェント名ヘッダ
- vm_agent_header: 仮想マシンタイプデバイスエージェントのエージェント名ヘッダ

フィルタ

- cluster_monitoring: クラスタ監視の有効化(1)または無効化(0)
- host_monitoring: 仮想マシンサーバ監視の有効化(1)または無効化(0)
- storage_monitoring: ストレージデバイス監視の有効化(1)または無効化(0)
- vm_monitoring: 仮想マシン監視の有効化(1)または無効化(0)

エンティティのリネーム

RENAME aaa TO bbb: エンティティのリネームルールで、必要なだけ定義することができます。

エンティティの除外

REJECT aaa

監視対象外エンティティのルールで、必要なだけ定義することができます。

プラグインの実行

システム cron で設定するには、次の行を /etc/crontab へ追加します。

```
/5 * * * * root "<path>/pandora_opennebula" "<path>/pandora_opennebula.conf">
/dev/null 2>&1
```

プラグインを手動実行すると、出力は次のようになります。

```
[root@valhalla ~]# ./pandora_opennebula pandora_opennebula.conf
[root@valhalla ~]# echo $?
0
```

IBM HMC

このプラグインは、HMC ハードウエア管理コンソールを通して IBM AIX 仮想環境を監視できます。このプラグインは、HMC システムによって AIX 環境に作成されたすべての論理パーティションから情報を収集します。各管理サーバ、論理パーティション、仮想 IO サーバごとに一つのエージェントを作成します。

SSH で情報を収集するために、プラグインは以下の 3つのモードを使えます。

1. スクリプト ssh_launcher.sh の利用をもとにしたもの
2. Net::SSH::Perl ライブラリをもとにしたもの
3. Net::SSH::Expect ライブラリをもとにしたもの

キャプチャされた情報を補完するために REST API に対してもクエリが実行されます。デフォルトは以下の通りです。

```
https://fqdn:12443/rest/api/{root_element}
```

必要条件

監視に必要なパラメータは次の通りです。

- HMC システムの認証に必要なユーザ名(読み出し専用)
  - ユーザは REST API への接続権限および HMC シェルへログインし(少なくとも)次のコマンドを実行する権限が必要です。
    - lssyscfg
    - lshwres
- ユーザのパスワード
- HMC の場所(FQDN/IP) (myhmc.mydomain など)
- HMC REST API のベース URL 例:

```
https://myhmc.mydomain:12443
```

プラグインにより生成されるモジュール

プラグインによって監視されるパラメータは次の通りです。(エレメントタイプでグループ化されます)

- *Current logical partitions* 展開されている現在の論理パーティション
- *Max logical partitions* 最大論理パーティション数
- *Max memory available* 未使用メモリ
- *Max memory installed* 最大搭載メモリ
- *Proc pool DefaultPool current proc units*
- *Proc pool DefaultPool max proc units*
- *Proc pool DevelopmentPool current proc units*
- *Proc pool DevelopmentPool max proc units*
- *Proc pool ProductionPool current proc units*
- *Proc pool ProductionPool max proc units*
- *Proc pool TestPool current proc units*
- *Proc pool TestPool max proc units*
- *Proc pool VIOPool current proc units*
- *Proc pool VIOPool max proc units*
- *Processor pools configured* 設定されているプロセッサプール
- *Processor units available* 未使用プロセッサユニット
- *Processor units installed* 搭載されているプロセッサユニット
- *State* 管理システムの状態
- *UUID* HMC API を照会するために使用されます
- *Virtual proc units max* 論理パーティションの最大仮想プロセッサユニット

LPAR:

- *Auto start* 自動起動設定の論理パーティション
- *LPAR type* 論理パーティションタイプ
- *LPAR UUID* HMC API を照会するために使用されます
- *Max memory* 最大メモリ
- *Max memory current* 未使用メモリ
- *Processor units available* 未使用プロセッサユニット
- *Processor units current* 搭載プロセッサユニット
- *RMC IP address* RMC IP アドレス
- *RMC state* LPAR の RMC の状態
- *State* 論理パーティションの状態
- *Virtual proc units* この LPAR に割り当てられた仮想プロセッサユニット

Virtual IO:

- *Auto start* 自動起動設定の論理パーティション
- *LPAR type* 論理パーティションタイプ
- *LPAR UUID* HMC API を照会するために使用されます
- *Max memory* 最大メモリ
- *Max memory current* 未使用メモリ
- *Processor units available* 未使用プロセッサユニット
- *Processor units current* 搭載プロセッサユニット
- *RMC IP address* RMC IP アドレス
- *RMC state* LPAR の RMC の状態

- *State* 論理パーティションの状態
- *Virtual proc units* この LPAR へ割り当てられている仮想プロセッサユニット

プラグイン設定

Pandora FMS サーバとの通信設定

- mode: データ転送モード“local” または “tentacle”
- tentacle_ip: Pandora サーバの IP アドレスtentacle モードの場合のみ利用。
- tentacle_port: Tentacle サーバのポート
- tentacle_opts: Tentacle の追加パラメータ
- tentacle_client: Tentacle クライアントのパス
- temp: テンポラリディレクトリ
- local_folder: データ転送モードが “local” の場合のパス

HMC へのアクセス設定

- hmc_host: HMC の IP または FQDN
- hmc_user: 読み出し権限のユーザ
- hmc_pass: パスワード
- as_agent_plugin: Pandora FMS エージェントで実行する場合(as_agent_plugin = 1)プラグインの出力は XML フォーマットで返されます。システムの cron で実行するかまたは、サーバプラグインとして実行する場合(as_agent_plugin = 0)は、標準出力に状態が返されます。

HMC エージェント設定

- agent_name: オプション。親エージェントの名前を設定します。デフォルトは 'hostname' です。
- agent_interval: エージェントの間隔。デフォルトは 300。
- agent_group: エージェントグループ。デフォルトは IBM

IBM HMC モジュールのカスタマイズ

- module_group: モジュールグループ。デフォルトは IBM
- module_interval: モジュールの間隔(倍率)。デフォルトは 1。
- module_tags: モジュールのタグ

エンティティのリネーム

エンティティのリネームにはrename というブロックを使います。

```
rename
 MyLPAR_NAME TO my new name
 MyLPAR_NAME2 TO my second new name
 rename_end
```

プラグインの実行

IBM AIX システムを HMC を通して監視するための Pandora プラグインは次のように展開します

as_agent_plugin パラメータを 1 に設定した場合(エージェントプラグインとしての実行):

```
module_plugin /usr/bin/perl pandora_hmc.pl pandora_hmc.conf
```

as_agent_plugin パラメータを 0 に設定した場合(サーバプラグインとしての実行):

```
# /etc/crontab
*/5 * * * * root /usr/bin/perl /root/hmc/pandora_hmc.pl
/root/vmware/pandora_hmc.conf
```

HPVM

HPVM プラグインの動作

このプラグインは、HPVM 仮想化サーバを監視できます。エージェントプラグインとして起動し、監視対象システム内に起動している各仮想マシンごとに、エージェントを生成します。

情報を収集するためにローカルコマンドを利用します。

プラグインに必要なもの

1. 監視対象のいコンピュータへの Pandora FMS エージェントのインストール
2. プラグインの実行権限を持ったユーザ
3. ユーザは、以下のように hpvvmstatus コマンドを実行する権限が必要です
    1. hpvmstatus
    2. hpvmstatus -X
    3. hpvmstatus -r -X

プラグインのインストール

Pandora FMS プラグインは、モジュールライブラリ からダウンロードします。コレクションとデプロイされた Pandora FMS エージェントを使用して実行をスケジュールしたり、ファイルの内容を不揮発性ディレクトリに抽出して、そこからシステムの cron を通じて実行したりできます。

```
unzip pandora_HPVM.zip
```

プラグイン設定

Pandora サーバとの通信設定

- mode: データ転送モード“local” または “tentacle”
- tentacle_ip: Pandora サーバの IP アドレス。データ転送モードが tentacle の場合のみ。
- tentacle_port: Tentacle サーバのポート
- tentacle_opts: Tentacle の拡張オプション
- tentacle_client: Tentacle クライアントのパス
- temp: テンポラリディレクトリ
- local_folder: データ転送モードが “local” の場合のパス

エージェント設定

- agent_name: オプション。エージェント名を設定します。デフォルトは 'hostname' です。
- agent_interval: エージェント間隔。デフォルトは 300。
- agent_group: エージェントグループ。デフォルトは HPVM

モジュールカスタマイズ

- module_group: モジュールグループ。
- module_interval: モジュールの間隔(倍率)。デフォルトは 1。
- module_tags: モジュールのタグ

プラグインの実行

Pandora FMS エージェントからのプラグイン実行では、エージェント設定ファイルに次の設定をします。

```
module_plugin /usr/bin/perl pandora_hpvm.pl pandora_hpvm.conf
```

手動でテストをするには、プラグインを設定し次のように起動します。

```
perl pandora_hpvm.pl pandora_hpvm.conf
```

Pandora FMS ドキュメント一覧に戻る