NTT Communications

Cloud^{n}

Monitoring
カスタムメトリクス
ご利用ガイド

Ver.1.0

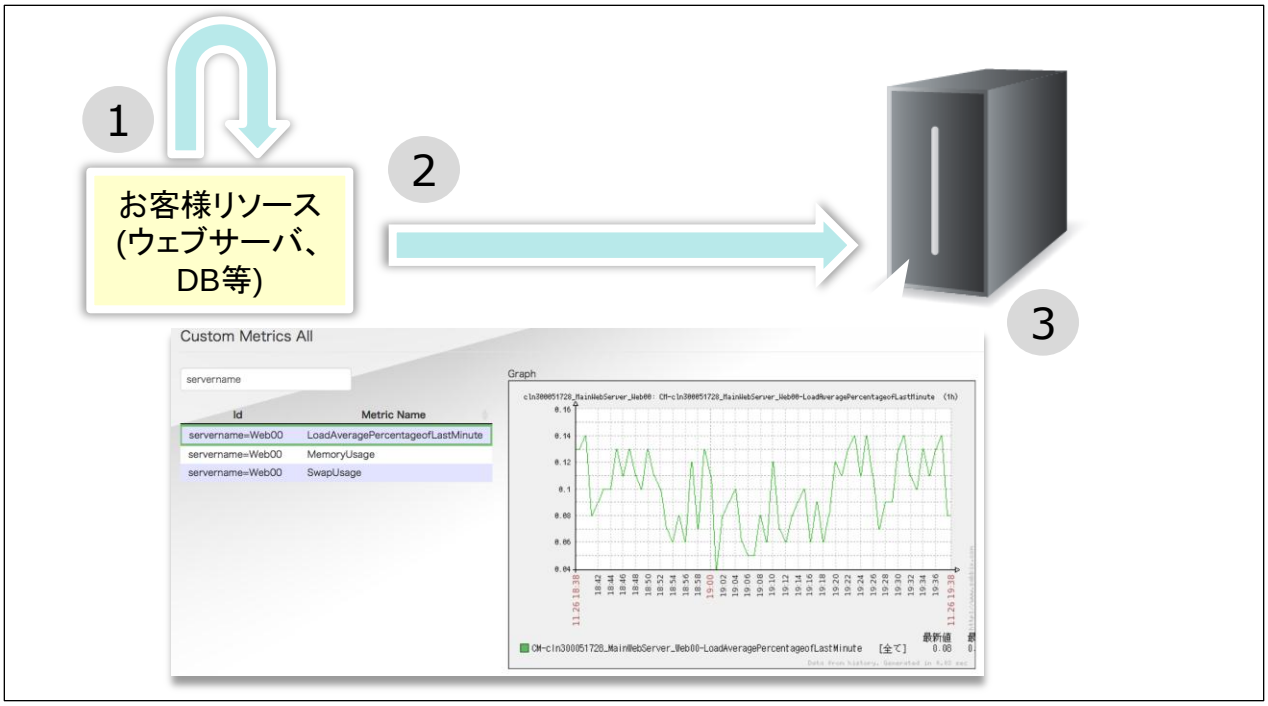
版数	編集日	変更内容
Ver.1.0	2014/04/21	初版作成

目次

1-1)カスタムメトリクスの概要

カスタムメトリクスとは、Cloudⁿ Monitoringが提供する、お客様が任意のメトリクスを設定可能とするサービスです。Monitoringがデフォルトで提供するメトリクス（ComputeのCPU使用率、ネットワークI/O等）以外のメトリクスデータもMonitoringサーバが蓄積することによって、お客様のすべてのメトリクスを一つのGUIで便利に一元管理することができます。

カスタムメトリクスの仕組みは以下の通りです：



1. 対象リソースから監視したいデータを抽出していただきます。例えば、ウェブサーバを運用されているお客様はーサーバが受け付けているHTTPリクエスト数や、メモリの利用状況を監視されたいかもしれません。こういったデータをスクリプト等で対象サーバから抽出していただきます。データ取得後、今度はそのデータをCloudⁿ Monitoringサーバに送っていただく必要があります。
2. 「PutMetricData」API*を利用し、抽出したデータをCloudⁿ Monitoringのサーバにプッシュしていただきます。より信頼性のあるメトリクスにするためには本「データ抽出→API実行」サイクルをcron等で自動的に定期実行されるよう設定されていることが望ましいです。
3. プッシュされてからおよそ1分後、お客様のカスタムメトリクスはCloudⁿ MonitoringのGUIに表示されます。なお、プッシュ頻度にもよりますが、ある程度メトリクスの「データ抽出→API実行」サイクルが繰り返されてからグラフを確認されることをお勧めします。ある程度メトリクスが蓄積されてからの方がデータの動向がわかります。

*本APIの詳細はMonitoringのAPIリファレンスマニュアルをご参照願います。

1-1)カスタムメトリクスの概要

「PutMetricData」APIはカスタムメトリクスの作成およびカスタムメトリクスのデータ登録の両方を同時に行う仕様のAPIであることご注意願います。

カスタムメトリクスは、2週間以上使用されていなければ自動的に削除されますことご注意ください。当該カスタムメトリクスに対し2週間以上データ投入がなければ使用されていないと判断されます。(API等でカスタムメトリクスを削除することはできません。)カスタムメトリクスのデータはデフォルトで提供しているメトリクスと同様、最近2週間のデータまで保持します。

2-1)サンプルスクリプトを使う

本項では、本ドキュメントと一緒に提供されているサンプルスクリプトの利用方法について説明します。

サンプルスクリプトはLinux上で動作することを想定された、メモリ・スワップ・平均負荷のメトリクスを取得するシェルです。デフォルトで本スクリプトを利用すれば3カスタムメトリクスができます。

1

Cloudⁿ のAPIはAmazon Web Services（通称：AWS）互換であるため、本例ではAWSが提供されているCloudWatch CLIツールを利用しながらサンプルスクリプトを動かします。そのため、まず以下のリンクより「CloudWatch CLI tool」をダウンロードしてください：

http://ec2-downloads.s3.amazonaws.com/CloudWatch-2010-08-01.zip

```
[user@webserver monitoring]$ wget http://ec2-downloads.s3.amazonaws.com/CloudWatch-2010-08-01.zip
```

ダウンロードされたzipファイルを監視対象サーバ内の適切なディレクトリに配置し、展開してください：

```
[user@webserver monitoring]$ unzip CloudWatch-2010-08-01.zip
```

上記の例ですと、「user」ユーザの~/cloudn/monitoringディレクトリに展開しました。

2

展開された資材の中に、「credential-file-path.template」というファイルがあります。本ファイルのコピーを取り、違う名前で保存してください。以下の例では、「credential-file」という名前でファイルを保存しました。

```
[user@webserver CloudWatch-1.0.13.4]$ cp credential-file-path.template credential-file
```

ファイルの内容を編集します。「<Write your AWS access ID>」と「<Write your AWS secret key>」と書かれている箇所を、お客様のCloudⁿサービス共通「アクセスキーID」および「秘密鍵」で代替してください。以下のようになるはずです：

```
AWSAccessKeyId=MJ3435RY7E465NG3B2PO
AWSSecretKey=cvdmKloJKHT7rWtRewvoefa0866JSRvZontwFiWM
```

内容が正しいことをご確認の上、本ファイルを保存し、閉じてください。

2-1)サンプルスクリプトを使う

3

次に、CLIツールを利用するために必要な環境変数を設定します。

以下の環境変数が必要となります：

JAVA_HOME・・・Javaのインストール先ディレクトリパス

AWS_CREDENTIAL_FILE・・・先ほど編集した「credential-file-path」ファイルまでのパス

AWS_CLOUDWATCH_HOME・・・展開した「CloudWatch CLIツール」の実行ディレクトリパス

AWS_CLOUDWATCH_URL・・・Cloud^{n} MonitoringのAPIエンドポイント

※**JAVA_HOME**、**AWS_CREDENTIAL_FILE**および**AWS_CLOUDWATCH_HOME**はお客様のサーバ環境に応じて適当に設定してください。以下は記入例です：

```
export JAVA_HOME=/usr
export AWS_CREDENTIAL_FILE=/home/user/cloudn/monitoring/CloudWatch-1.0.13.4/credential-file
export AWS_CLOUDWATCH_HOME=/home/user/cloudn/monitoring/CloudWatch-1.0.13.4/
export PATH=$PATH:$AWS_CLOUDWATCH_HOME/bin
export AWS_CLOUDWATCH_URL=https://mon-api.jp-e1.cloudn-service.com/
```

上記の環境変数をお客様の~/.bashrcファイルに設定いただければサーバのどこからでもCloudWatch CLIを実行することができますが、本項ではひとまず”sample_script.sh”サンプルスクリプト内にこれらパラメータを設定します。

4

“sample_script.sh” を同サーバ上の任意のディレクトリに置いてください。以下の例では「user」ユーザの~/workディレクトリ内に置きました。

```
[user@webserver work]$ ls
sample_script.sh
```

ファイルを開き、4から8行目に適当な環境変数を設定してください。以下のようになるはずです：

```
#!/bin/bash

# Enter the environment variables necessary for CLI tool to function:
export JAVA_HOME=/usr
export AWS_CREDENTIAL_FILE=/home/user/cloudn/monitoring/CloudWatch-1.0.13.4/credential-file
export AWS_CLOUDWATCH_HOME=/home/user/cloudn/monitoring/CloudWatch-1.0.13.4/
export PATH=$PATH:$AWS_CLOUDWATCH_HOME/bin
export AWS_CLOUDWATCH_URL=https://mon-api.jp-e1.cloudn-service.com/
```

2-1)サンプルスクリプトを使う

5

カスタムメトリクスを作成するには、①メトリクス名、②ネームスペースおよび③ディメンションの3点を定義する必要があります。ネームスペースとディメンションはサンプルスクリプトの11と12行目で編集できます。メトリクス名は取得しているデータに応じて適切な名前があらかじめ付けてあるが、編集したい場合はそれぞれ32、34、36（とコメントアウトされているが37と38）行目に書かれています。

以下、ネームスペースとディメンションの記入例です：

```
# Define the namespace and dimension name-value pair for the resource you will be
creating custom metrics for:
NAMESPACE="MainWebServer"
DIMENSIONS="servername=Web00"
```

内容が正しいことをご確認の上、ファイルを保存し、閉じてください。

6

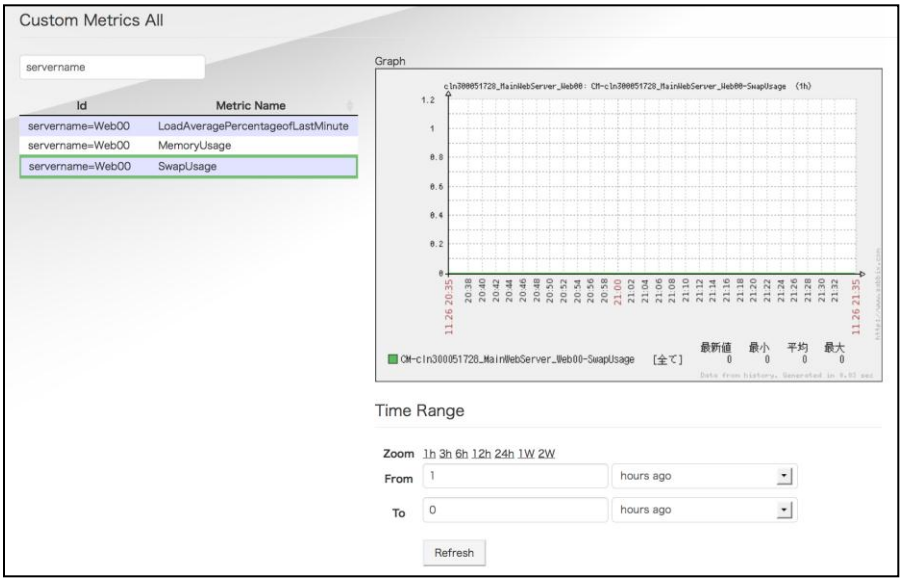
ファイルを実行可能にするためにパーミッションを変更し、実行してください。

```
[user@webserver work]$ ./sample_script.sh
```

2-1)サンプルスクリプトを使う

7

APIまたはGUI経由で3つのカスタムメトリクスが作成されたこと確認してください。



ーカスタムメトリクスの作成から初メトリクス表示まで1分程度かかることがあります。

8

サンプルスクリプトを定期的かつ自動的に実行するために、cronでジョブ設定します。cronジョブを設定するためには*crontab –e*コマンドを使用します。

```
[user@webserver ~]$ crontab -e
```

既にジョブがなければ空の画面が表示されます。cronの形式に則って、”sample_script.sh”が定期実行されるようにファイルを編集します。以下の例では”sample_script.sh”を毎5分実行するよう、「user」ユーザのcronを設定しました:

```
CUSTOM_METRICS_DIR=“/home/user/work”

*/5 * * * * $CUSTOM_METRICS_DIR/sample_script.sh
```

内容が正しいことをご確認の上、ファイルを保存し、閉じてください。正しく実行されていることを*less /var/log/cron* 等のコマンドでご確認ください。ある程度の時間が経ちましたら、お客様のサーバのメモリ・スワップ・平均負荷メトリクスデータが蓄積され始めます。

2-2)さらに活用するためには？

前項では、カスタムメトリクスのある一つの利用方法をご紹介させていただきました。しかし、サンプルスクリプトはあくまでも一例です。カスタムメトリクスを使って他にも色んなメトリクスを管理することができます。

監視したい項目をすべてCloud^{n} Monitoringに集めていただくことによって、リソース一元管理の実現と、それによる運用負担の軽減を実現することができます。

カスタムメトリクスを利用すれば、Cloud^{n}のリソースに限らず、数値化できるデータすべてを管理することができます。Cloud^{n} Monitoringのエンドポイントに届く先であれば、どんなデータでも溜めることができます。