ZABBIX 2020 Conference JAPAN

HTTPエージェントを利用したクラウドAPI利用

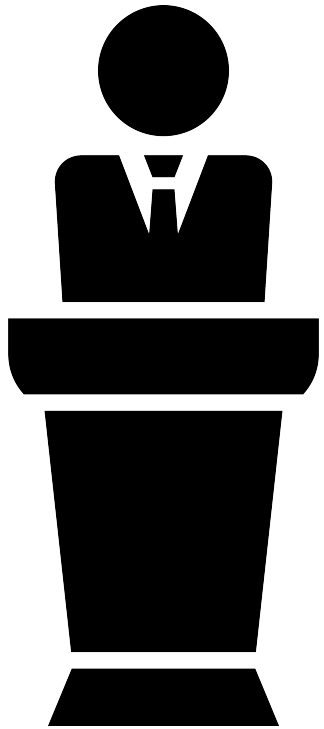
#ZabConfJp2020

自己紹介

名前： 比嘉 啓太
職種： サポートエンジニア
Zabbix歴： 2012年頃からZabbixを使用
2018年Zabbix Japan入社

Agenda

1. HTTPエージェントとは
2. 保存前処理と依存アイテム
3. HTTPエージェントをAWSで使用する
4. まとめ



HTTPエージェントとは

ZABBIX 2020 Conference JAPAN

#ZabConfJp2020

HTTPエージェントとは

- Zabbixサーバからhttp/https接続を使用して値の取得
  - 監視対象でZabbixAgentを使用する必要が無い

- 内部の処理はcURL(libcurl)を使用
- 取得するデータを保存前処理で加工することが可能

- 参考情報:
  - Zabbix Documentation: 17 HTTP AGENT
  - 公式テンプレート「Template App Apache by HTTP」など

HTTPエージェントの例

Name: Apache: Get status
Type: HTTP agent
Key : apache.get_status
URL: {$APACHE.STATUS.SCHEME}://{HOST.CONN}:{$APACHE.STATUS.PORT}/{$APACHE.STATUS.PATH}
実際のURLは http://127.0.0.1:80/server-status?auto となる

タイムスタンプ	値
2020/11/10 14:57:41	{"Date":"Tue, 10 Nov 2020 05:57:41 GMT","Server":"Apache/2.4.37 (centos)","Length":1170,"Type":"text/plain; charset=ISO-8859-1","ServerVersion":"Apache/2.4.37 (centos)","ServerMPM":"event","Server Built":"Sep 15 2020 15:41:16","CurrentTime":"Tuesday, 10-Nov-2020 05:57:41 UTC","RestartTime":"Tuesday, 10-Nov-2020 05:53:37 UTC","ParentServerConfigGeneration":1,"ParentServerMPMGeneration":0,"ServerUptimeSeconds":244,"ServerUptime":"4 minutes 4 seconds","Load1":0,"Load5":0,"Load15":0.01,"Total Accesses":110,"Total kBytes":744,"Total Duration":818,"CPUUser":0.13,"CPUSystem":0.11,"CPUChildrenUser":0,"CPUChildrenSystem":0,"CPULoad":0.0983607,"Uptime":244,"ReqPerSec":0.45082,"BytesPerSec":3122.36,"BytesPerReq":6925.96,"DurationPerReq":7.43636,"BusyWorkers":1,"IdleWorkers":99,"Processes":4,"Stopping":0,"ConnsTotal":3,"ConnsAsyncWriting":0,"ConnsAsyncKeepAlive":1,"ConnsAsyncClosing":0,"Scoreboard":"_____W_____.....","Workers":{"waiting":99,"starting":0,"reading":0,"sending":1,"keepalive":0,"dnslookup":0,"closing":0,"logging":0,"finishing":0,"cleanup":0,"slot":300}}

Zabbixが取得する値


```
{
  "Date": "Tue, 10 Nov 2020 06:00:41 GMT",
  "Server": "Apache/2.4.37 (centos)",
  "Length": 1172,
  "Type": "text/plain; charset=ISO-8859-1",
  "ServerVersion": "Apache/2.4.37 (centos)",
  "ServerMPM": "event",
  "Server Built": "Sep 15 2020 15:41:16",
  "CurrentTime": "Tuesday, 10-Nov-2020 06:00:41 UTC",
  "RestartTime": "Tuesday, 10-Nov-2020 05:53:37 UTC",
  "ParentServerConfigGeneration": 1,
  "ParentServerMPMGeneration": 0,
  "ServerUptimeSeconds": 424,
  "ServerUptime": "7 minutes 4 seconds”
```


〜〜〜以下略〜〜〜

保存前処理と依存性アイテム

ZABBIX 2020 Conference JAPAN

#ZabConfJp2020

保存前処理とは

- Zabbixが取得したアイテムデータを様々な形で加工する
- JSONやXML Xpathといった構造化データの編集が容易

保存前処理の種類

テキスト	正規表現、置換、前後文字列削除、末尾文字列削除、先頭文字列削除
構造データ	XML Xpath、JSON Path、CSVからJSON
計算	乗数
変化	差分、1秒あたりの差分
数値変換	論理値から10進数、8進数から10進数、16進数から10進数
カスタムスクリプト	JavaScript
その他	バリデーション、絞り込み、Prometeus等

- アイテムの保存前処理から設定が可能
- 設定時にはテストも実施可

保存前処理の実行前

http://127.0.0.1:80/server-status?autoの場合

ServerVersion: Apache/2.4.37 (centos)

ServerMPM: event

Server Built: Sep 15 2020 15:41:16

CurrentTime: Tuesday, 10-Nov-2020 07:53:19 UTC

RestartTime: Tuesday, 10-Nov-2020 05:53:37 UTC

ParentServerConfigGeneration: 1

ParentServerMPMGeneration: 0

ServerUptimeSeconds: 7182

ServerUptime: 1 hour 59 minutes 42 seconds

Load1: 0.00

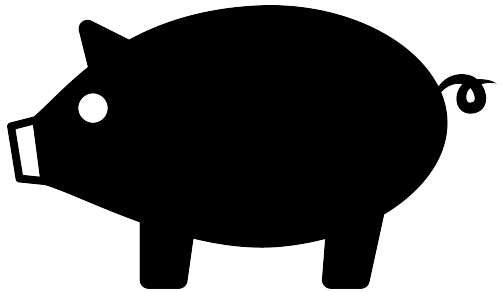
Load5: 0.00

Load15: 0.00

Total Accesses: 1992

Total kBytes: 20662

Total Duration: 16499



アイテム　保存前処理

保存前処理の設定

	名前	パラメータ	失敗時のカスタマイズ	アクション
1:	JavaScript	// Convert Apache status to JSON…		テスト　削除

追加　　すべてのテスト

更新　複製　テスト　削除　キャンセル

JavaScript

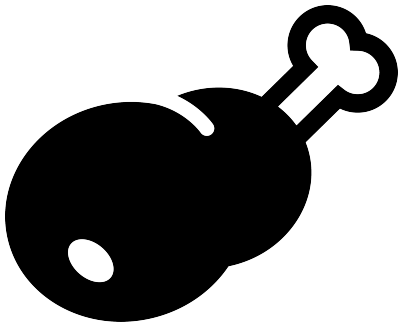
function (value) {

```
// Convert Apache status to JSON
var lines = value.split('\n');
var output = {},
    workers = {
        '_': 0, 'S': 0, 'R': 0, 'W': 0,
        'K': 0, 'D': 0, 'C': 0, 'L': 0,
        'G': 0, 'I': 0, '.': 0
    };

// Get all "Key: Value" pairs as an object
for (var i = 0; i < lines.length; i++) {
    var line = lines[i].match(/([A-z0-9 ]+): (.*)/);

    if (line !== null) {
        output[line[1]] = isNaN(line[2]) ? line[2] : Number(line[2]);
    }
}

// Multiversion metrics
output.ServerUptimeSeconds = output.ServerUptimeSeconds || output.Uptime;
output.ServerVersion = output.Server || output.ServerVersion;

// Parse "Scoreboard" to get worker count.
if (typeof output.Scoreboard === 'string') {
    for (var i = 0; i < output.Scoreboard.length; i++) {
        var char = output.Scoreboard[i];

        workers[char]++;
    }
}

// Add worker data to the output
output.Workers = {
    waiting: workers['_'], starting: workers['S'], reading: workers['R'],
    sending: workers['W'], keepalive: workers['K'], dnslookup: workers['D'],
```

保存前処理の実行後

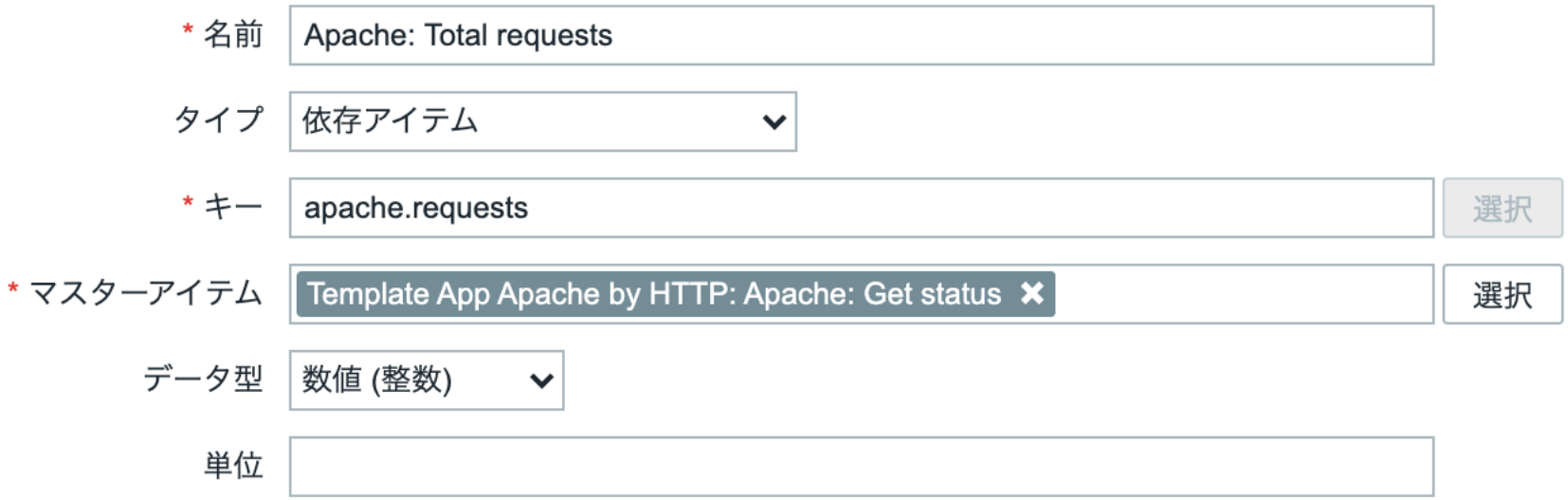
ServerVersion: Apache/2.4.37 (centos)
ServerMPM: event
Server Built: Sep 15 2020 15:41:16
CurrentTime: Tuesday, 10-Nov-2020 07:53:19 UTC
RestartTime: Tuesday, 10-Nov-2020 05:53:37 UTC
ParentServerConfigGeneration: 1
ParentServerMPMGeneration: 0
ServerUptimeSeconds: 7182
ServerUptime: 1 hour 59 minutes 42 seconds
Load1: 0.00
Load5: 0.00
Load15: 0.00
Total Accesses: 1992
Total kBytes: 20662
Total Duration: 16499

{
"Date": "Tue, 10 Nov 2020 06:00:41 GMT",
"Server": "Apache/2.4.37 (centos)",
"Length": 1172,
"Type": "text/plain; charset=ISO-8859-1",
"ServerVersion": "Apache/2.4.37 (centos)",
"ServerMPM": "event",
"Server Built": "Sep 15 2020 15:41:16",
"CurrentTime": "Tuesday, 10-Nov-2020 06:00:41 UTC",
"RestartTime": "Tuesday, 10-Nov-2020 05:53:37 UTC",
"ParentServerConfigGeneration": 1,
"ParentServerMPMGeneration": 0,
"ServerUptimeSeconds": 424,
"ServerUptime": "7 minutes 4 seconds",
"Load1": 0.01,



依存性アイテムとは

- マスターアイテムが取得したアイテムを使用する
- 取得済みのJSONなどのアイテムから項目別に取得が可能



{

"Date": "Tue, 10 Nov 2020 19:00:41 GMT",

～～～中略～～～

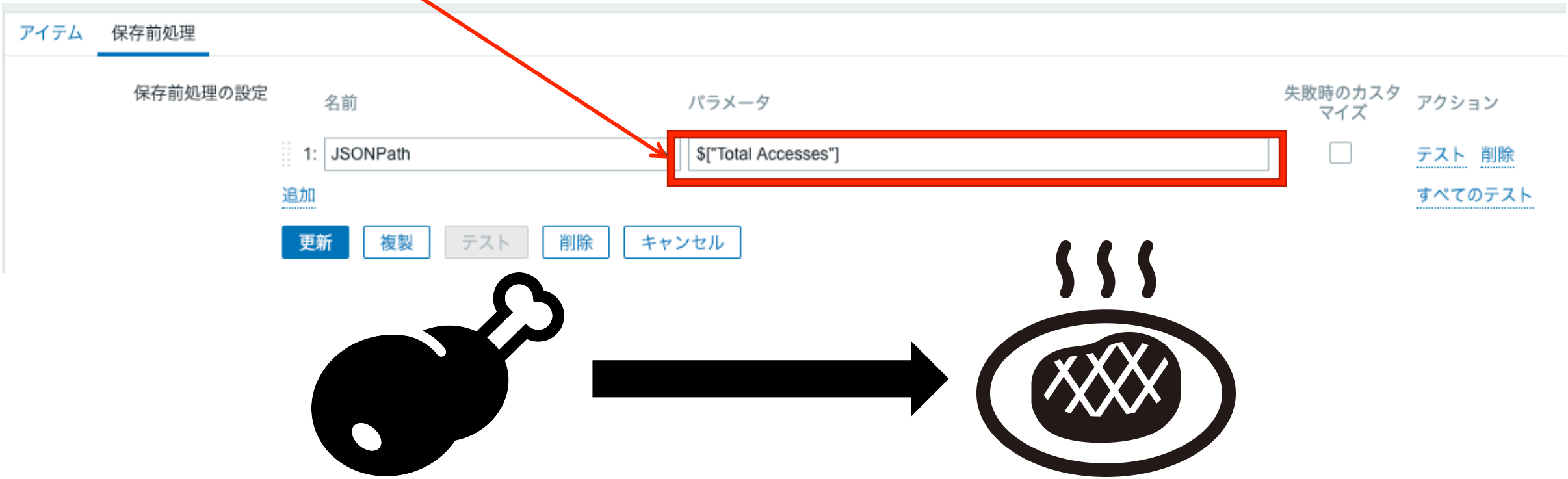
"Load15": 0,

"Total Accesses": 5922,

“Total kBytes”: 1012,

HTTPエージェントアイテム

Apache: Get statusで取得したJSONから欲しい値を抽出



JSONやXMLXpathなどの
取得した他のアイテムを
基に様々なアイテムを作成できる

HTTPエージェントをAWSで使用する

ZABBIX 2020 Conference JAPAN

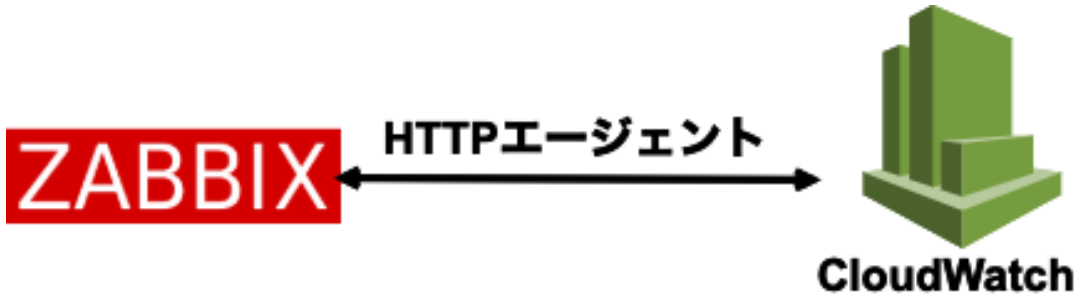
#ZabConfJp2020

一般的な取得方法

- Zabbixサーバから外部スクリプトなどを使用して値を取得する
  - BashなどでCLIを呼び出す、Python等を使用しSDKで呼び出す

試験環境

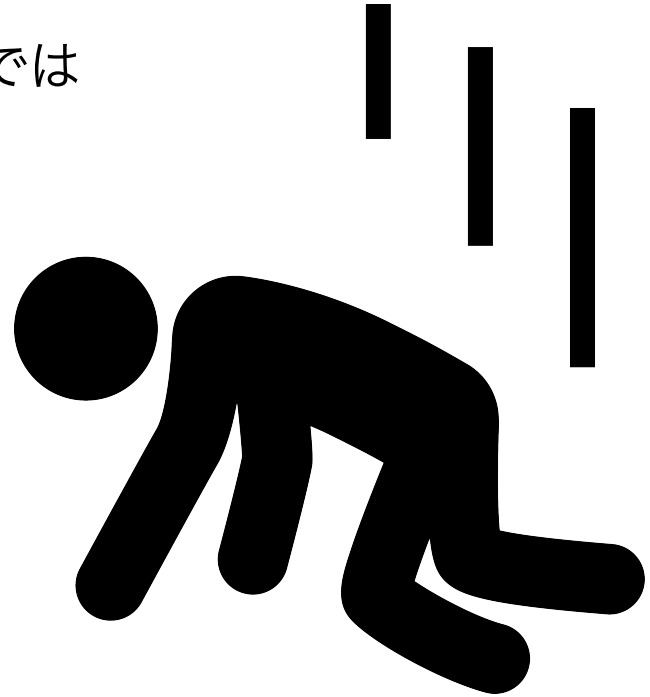
- ZabbixServer(5.0.2)
- AWSの環境(CloudWatchへのアクセス)
- AWS SDKは使用しない



結論

HTTPエージェント機能のみで
CloudWatchから値をとることはできない

※5.0まででは



検証で詰まった点

- AWSのAPIをHTTPリクエストで実行する場合AWSが送信元を特定するための署名が必要となる
- 署名を取得するにはリクエストのハッシュが必要

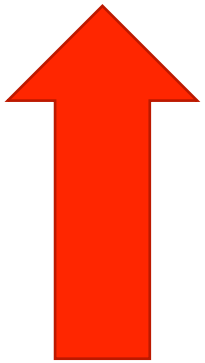
- 参考ドキュメント：　AWS API リクエストの署名
  Making API Requests

API リクエストの署名とは

1. 正規リクエストの作成
2. 署名文字列の作成
3. 署名の計算
4. HTTPリクエストへ署名を追加

参考URL：
署名バージョン 4 署名プロセス
https://docs.aws.amazon.com/ja_jp/general/latest/gr/signature-version-4.html

1.正規リクエストの作成



HTTPエージェントでは
リクエスト作成の処理からできない

なぜ正規リクエストが作成できないか

- 正規リクエストの作成では
  リクエストのハッシュ化の処理が必要

- HTTPエージェントではハッシュ化はできない

- 正規リクエストが作れない

```
CanonicalRequest =
        HTTPRequestMethod + '¥n' +
        CanonicalURI + '¥n' +
        CanonicalQueryString + '¥n' +
        CanonicalHeaders + '¥n' +
        SignedHeaders + '¥n' +
         HexEncode(Hash(RequestPayload))
```

Action=ListMetrics

Version=2010-08-01

X-Amz-Algorithm=AWS4-HMAC-SHA256

X-Amz-Credential=AKIABCDEFGHIJKLMNOPQ%2F20201114%2Fap-northeast-1%2Fmonitoring%2Faws4_request

X-Amz-Date=20201114T030050Z

X-Amz-SignedHeaders=host

CANONICAL REQUEST

GET /doc/2010-08-01/ Action=ListMetrics&Version=2010-08-01&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIABCDEFGHIJKLMNOPQ%2F20201114%2Fap-northeast-1%2Fmonitoring%2Faws4_request&X-Amz-Date=20201114T030534Z&X-Amz-SignedHeaders=host
host:monitoring.ap-northeast-1.amazonaws.com host
e3b0c44298fc1c149afbf4c11926fb92427ae41e4649b934ca495991b7852b855

CANONICAL_REQUEST

GET /doc/2010-08-01/ Action=ListMetrics&Version=2010-08-01&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIABCDEFGHIJKLMNOPQ%2F20201114%2Fap-northeast-1%2Fmonitoring%2Faws4_request&X-Amz-Date=20201114T030534Z&X-Amz-SignedHeaders=host
host:monitoring.ap-northeast-1.amazonaws.com host
e3b0c44298fc1c149afbf4c11926fb92427ae41e4649b934ca495991b7852b855

SHA256によるHash化

CANONICALREQUEST_HASH

7f9b829295273c5d84739b7c09c10eadcf5b5eb68d9fb2b748135cec2080c442

CANONICAL_REQUEST

GET /doc/2010-08-01/ Action=ListMetrics&Version=2010-08-01&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIABCDEFGHIJKLMNOPQ%2F20201114%2Fap-northeast-1%2Fmonitoring%2Faws4_request&X-Amz-Date=20201114T030534Z&X-Amz-SignedHeaders=host
host:monitoring.ap-northeast-1.amazonaws.com host
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855

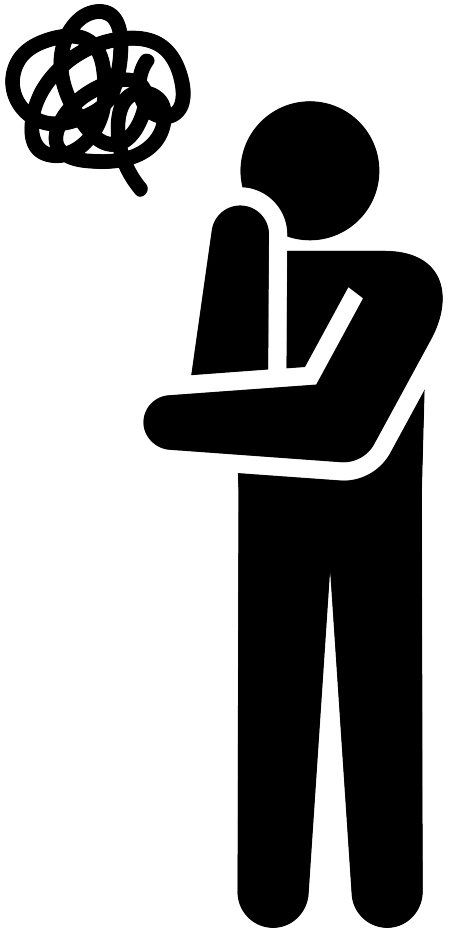
SHA256によるHash化

CANONICALREQUEST_HASH

7f9b829295273c5d84739b7c09c10eadcf5b5eb68d9fb2b748135cec2080c442

これがHTTPエージェントではできない

他の手段はないか？



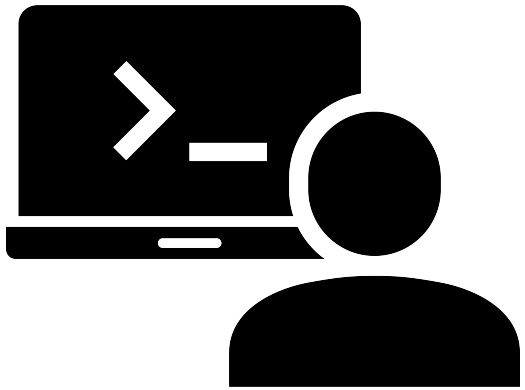
Lambda経由でCloudWatchを使用する

- ZabbixServer(5.0.2)
- AWSの環境
  - APIGateway
  - CloudWatch
  - Lambda

5.4のRoadMapではクラウドの対応が計画中(AWS,Azure,GCP等)

Out of the box monitoring and alerting

- Monitoring of Azure, AWS and Google Cloud Platform services
- Monitoring of Kubernetes clusters



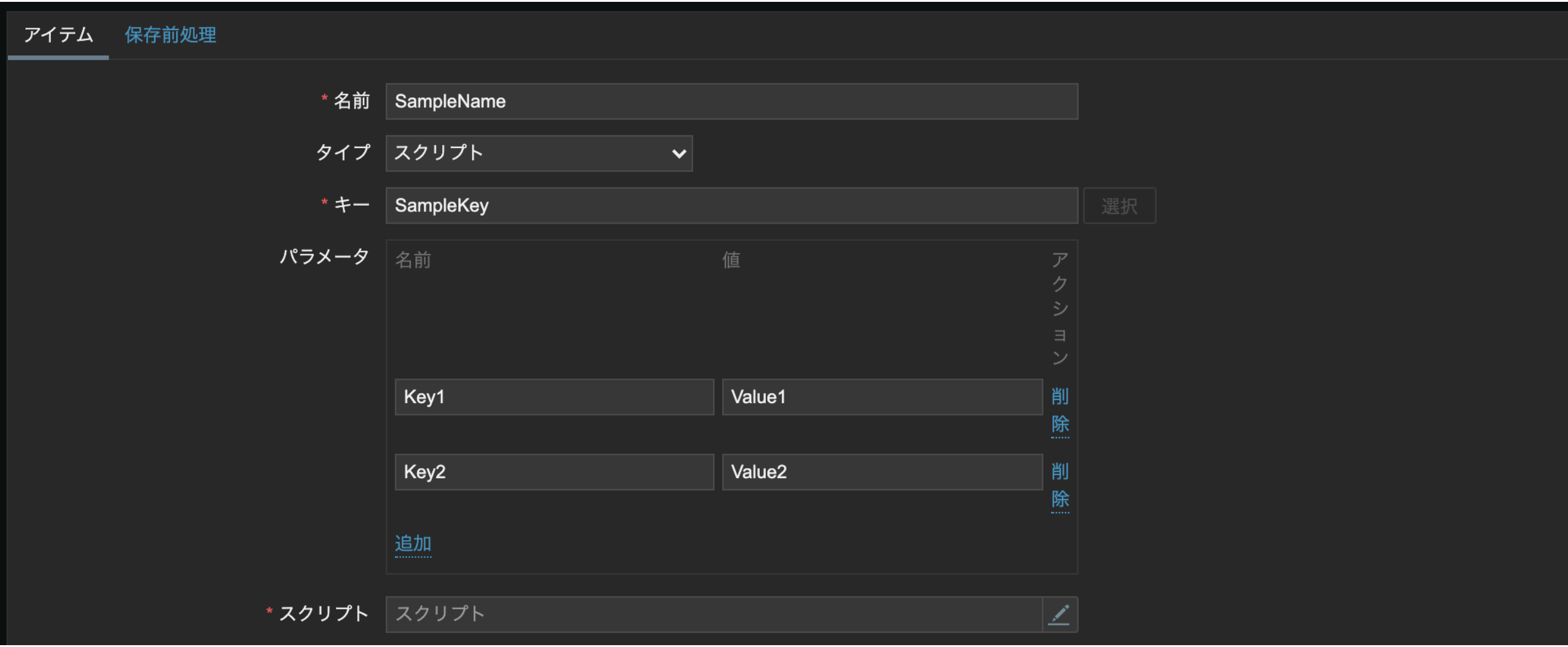
おまけ（HTTPエージェント以外）

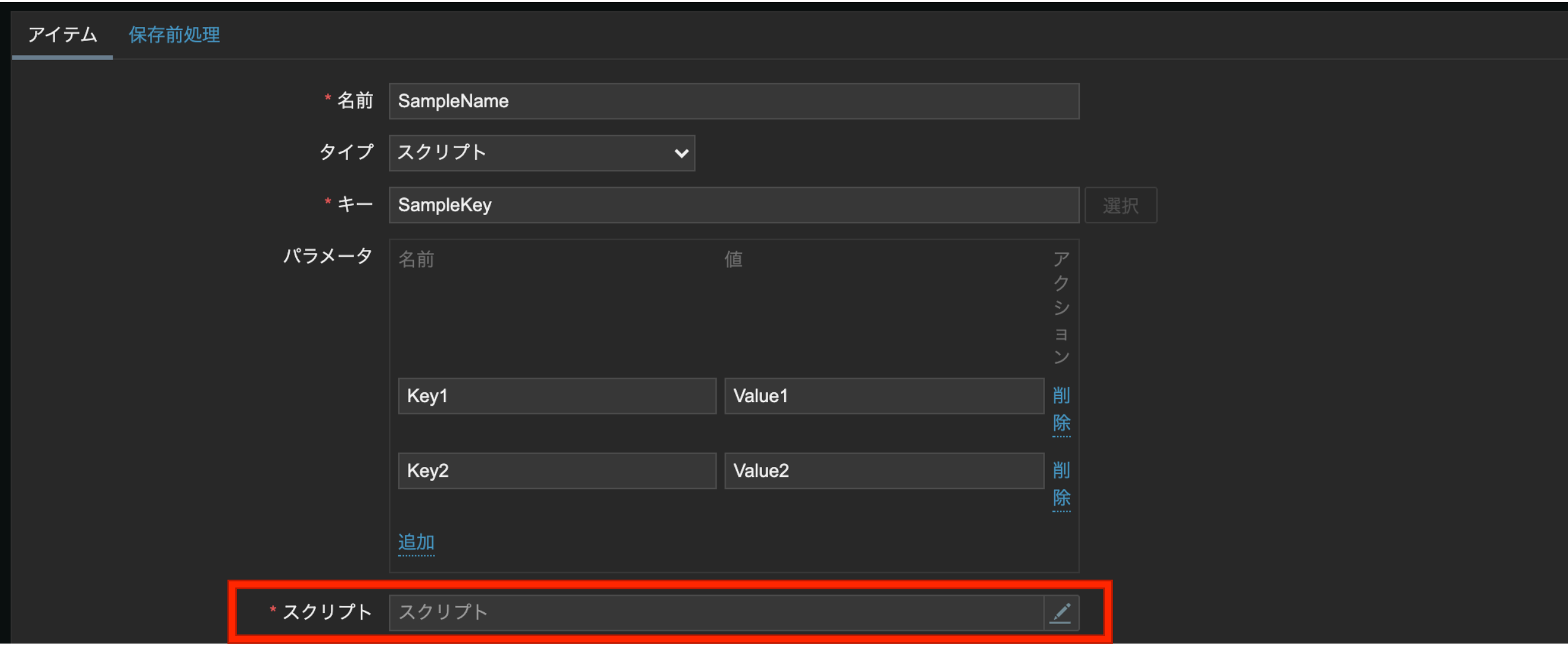
ZABBIX 2020 Conference JAPAN

#ZabConfJp2020

5.2ではJavaScriptが使える

- zabbix_jsというコマンドがある
- Zabbixで使用するJavaScriptはDuktapeと呼ばれるエンジン
- 標準ライブラリ以外を使用することができない


アイテム
保存前処理
* 名前
SampleName
タイプ
スクリプト
* キー
SampleKey
選択
パラメータ
名前
値
アクション
Key1
Value1
削除
Key2
Value2
削除
追加
* スクリプト
スクリプト


アイテム
保存前処理
* 名前
SampleName
タイプ
スクリプト
* キー
SampleKey
選択
パラメータ
名前
値
アクション
Key1
Value1
削除
Key2
Value2
削除
追加
* スクリプト
スクリプト

```
    "canonicalurl": "/doc/2010-08-01/", \
    "canonicalquerystring": {\
      "Action":"Action=ListMetrics",\
      "Version":"Version=2010-08-01",\
      "XAmzAlgorithm":"X-Amz-Algorithm=AWS4-HMAC-SHA256",\
      "XAmzCredential":"X-Amz-Credential=",\
      "XAmzDate":"X-Amz-Date=",\
      "XAmzSignedHeaders":"X-Amz-SignedHeaders="\
    }\
  }';

//  Zabbix.Log(4, "value:"+value);
//  Zabbix.Log(4, "Date.now()"+format_date);

  var params = JSON.parse(value);
  var CanonicalRequest;
  var rightNow = new Date();
  var CanonicalUrl;
  var CanonicalQueryString = {},
      QueryString;
  var CanonicalHeaders = {};
  var RequestPayload;
  var payload_hash;
  var ret;

  //SetDate
  params.Expiration = rightNow.toISOString().replace(/\.\d+/, "");
  params.canonicalquerystring.XAmzDate += rightNow.toISOString().replace(/\.\d+/, "");

  //Create X-Amz-Credential
  params.canonicalquerystring.XAmzCredential +=
    //accessKeyId
    params.AccessKeyId + "/" +
      //amz_scope
      rightNow.getFullYear() + "" + rightNow.getMonth() + "" + rightNow.getDate() + "/" +
      params.region + "/" + params.credential_scope;
  params.canonicalquerystring.XAmzCredential=encodeURI(params.canonicalquerystring.XAmzCredential);

  params.canonicalquerystring.XAmzSignedHeaders+=params.signedHeaders;

  // Set
  ret = "GET"
  //Set URI
  ret +=  "\n" + params.canonicalurl;
```

CANONICAL_REQUEST

GET /doc/2010-08-01/ Action=ListMetrics&Version=2010-08-01&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIABCDEFGHIJKLMNOPQ%2F20201114%2Fap-northeast-1%2Fmonitoring%2Faws4_request&X-Amz-Date=20201114T030534Z&X-Amz-SignedHeaders=host
host:monitoring.ap-northeast-1.amazonaws.com host
e3b0c44298fc1c149afbf4c8996fd92427ae43e4649b934ca495991b7852b855

SHA256によるHash化

CANONICALREQUEST_HASH

7f9b82929d273c5d84739d7c09c10eadcf5d5eb68d9fb2b748135cec2089c442

ここまではできた

署名文字列の計算

kSecret = SECRET_ACCESSKEY
kDate = HMAC("AWS4" + kSecret, Date)
kRegion = HMAC(kDate, Region)
kService = HMAC(kRegion, Service)
kSigning = HMAC(kService, "aws4_request")

HMAC-SHA256によるHash化

X-Amz-Signature
e2dd8ca9ca19330a706f5ecd98f17354d30e4509d4fa3aa5b79872439eabdb1a

この処理までは間に合いませんでした

まとめ

ZABBIX 2020 Conference JAPAN

#ZabConfJp2020

まとめ

- 現在のHTTPエージェントでクラウド系で使用するのは難しい

- 実際に監視する対象の仕様に依存する
  - 認証周りのToken取得が容易でない場合は別の手段と合わせて使用する必要がある
  - APIGateway-Lambdaのように前1段かませることによって実装は容易になる

- 5.2ではJSが使用できる様になったので取得が行えるかもしれない
  - こちらは最後まで検証が行えなかった

HTTPエージェント自体は便利

- サンプルで使用した様に特定のURLからデータを取得
- RESTful APIな環境では容易に使用ができる


Thank you
ZABBIX 2020
Conference
JAPAN
#ZabConfJp2020