SRA OSS, INC.

Zabbix 4.4 解説

Zabbix Conference Japan 2019
2019/11/15

SRA OSS, Inc. 日本支社
赤松 俊弘

ZABBIX
PREMIUM PARTNER

whoami

自己紹介

- ☑ 赤松 俊弘（Toshihiro Akamatsu）
- ☑ SRA OSS, Inc. 日本支社
  OSS 事業本部 基盤技術グループ

職務

- ☑ PostgreSQL 以外の OSS 全般の技術サポート、構築
- ☑ 主に Zabbix を担当

Zabbix 4.4 概要

Zabbix 4.4 概要

- ☑ 2019/10/07 リリース
- ☑ ポイントリリース
  - ☑ **フルサポート６カ月**
  - ☑ リミテッドサポート１ヶ月
  - ☑ 延長サポートなし

公式サポート環境の追加

OS

Debian 10

Raspbian 10

Mac OS/X

SUSE Linux Enterprise Server 15

Red Hat Enterprise Linux 8

MSI for Windows agent

パッケージ提供バージョン

- ☑ 3.0 LTS
- ☑ 4.0 LTS
- ☑ 4.2
- ☑ 4.4

公式サポート環境の追加

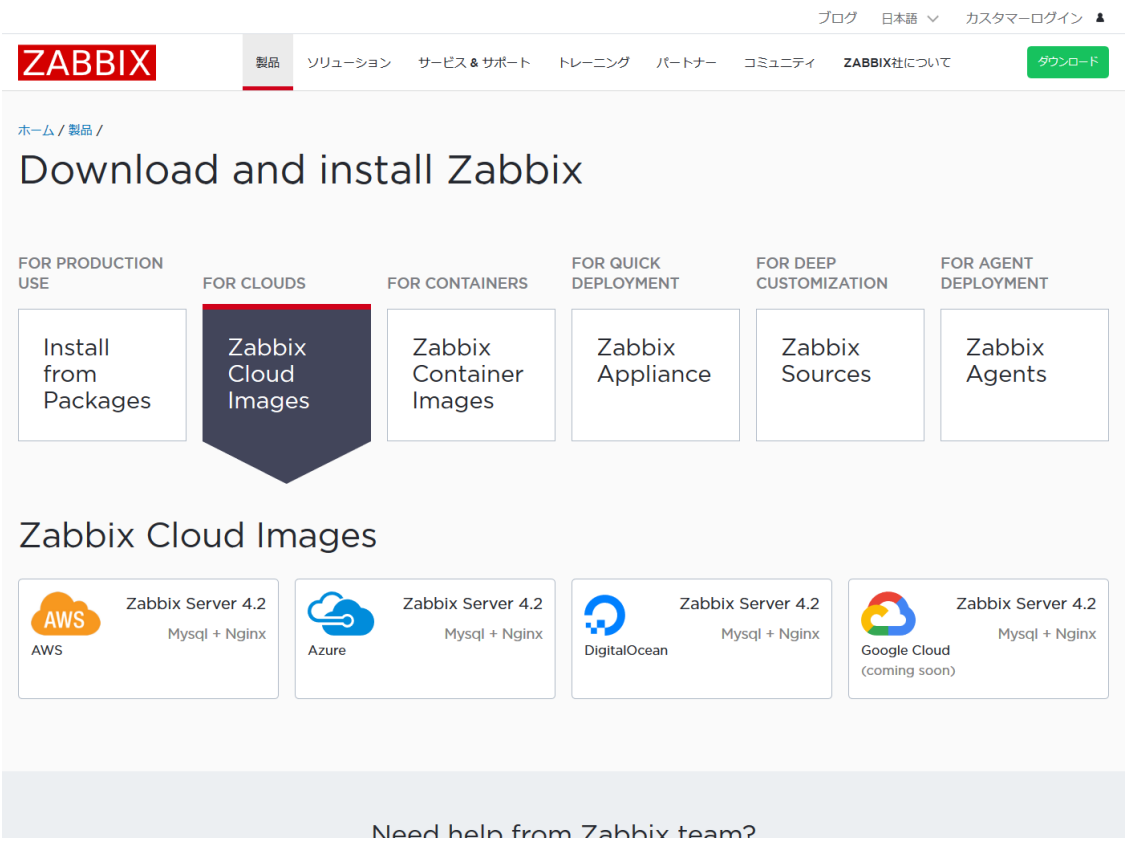
クラウドイメージ

AWS

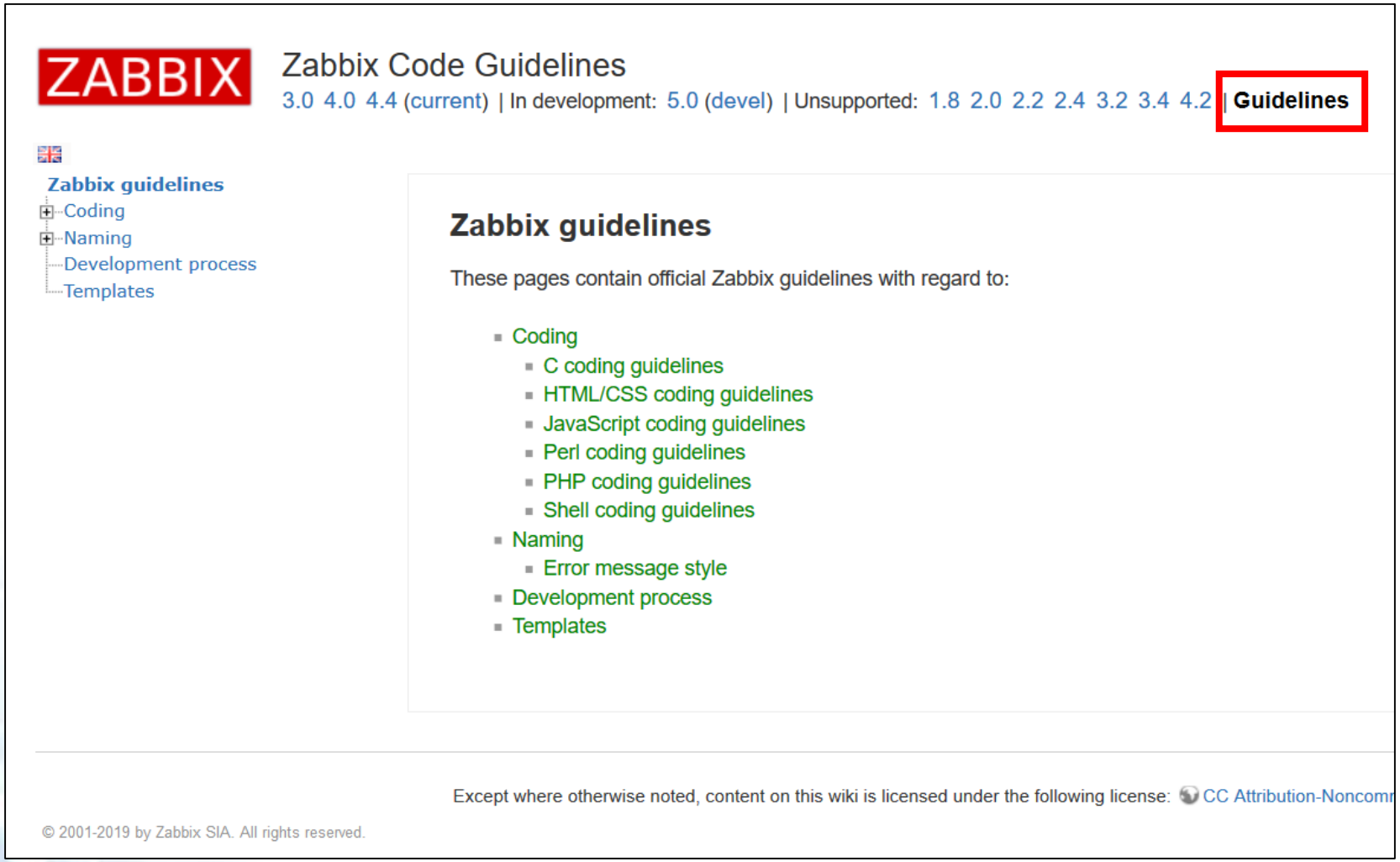
DigitalOcean

Microsoft Azure

Google Cloud

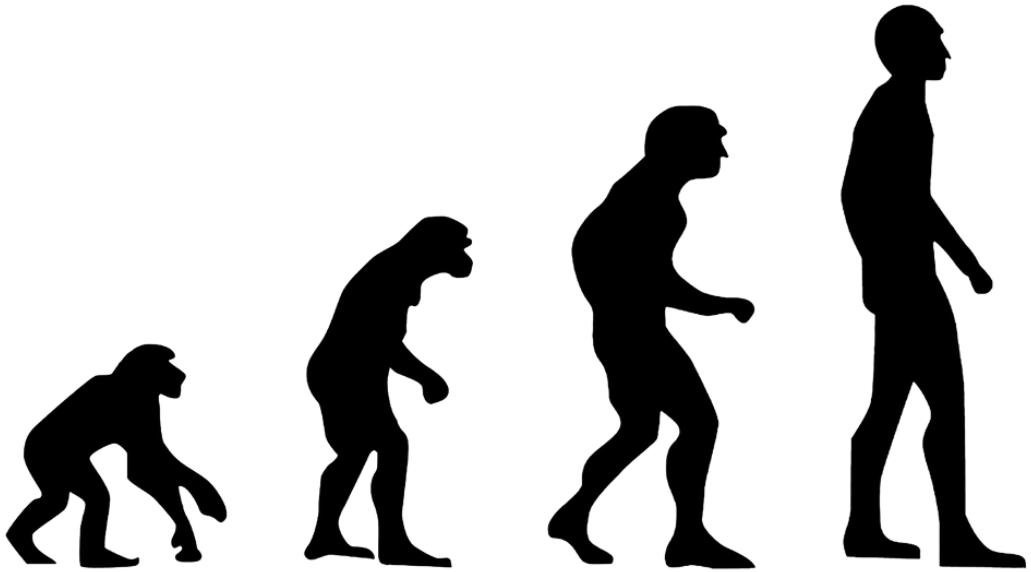


各種ガイドラインの公開



Zabbix 4.4 の新機能

- ☑ 次世代の Zabbix Agent
- ☑ Webhook によるアラート
- ☑ TimescaleDB の公式サポート
- ☑ グラフでの集約関数
- ☑ LLD アイテムの追加
- ☑ 保存前処理の追加
- ☑ Frontend の改善
- ☑ etc...



次世代の Zabbix Agent

その名も

Zabbix Agent 2

詳細は…

Magic of the new Zabbix Agent

Alexey Pustovalov

15:30 - 16:10

次世代の Zabbix Agent

- ☑ Go 言語で開発された新しい Zabbix Agent
- ☑ 現状は実験的なサポート段階

サポートプラットフォーム

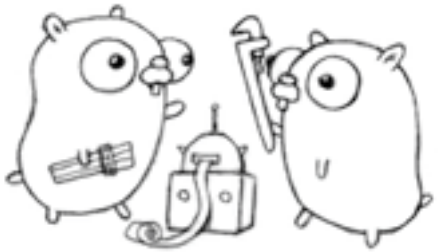
以下はパッケージあり

- ☑ RHEL/CentOS 8
- ☑ SLES 15 SP1+
- ☑ Debian 9, 10
- ☑ Ubuntu 18.04

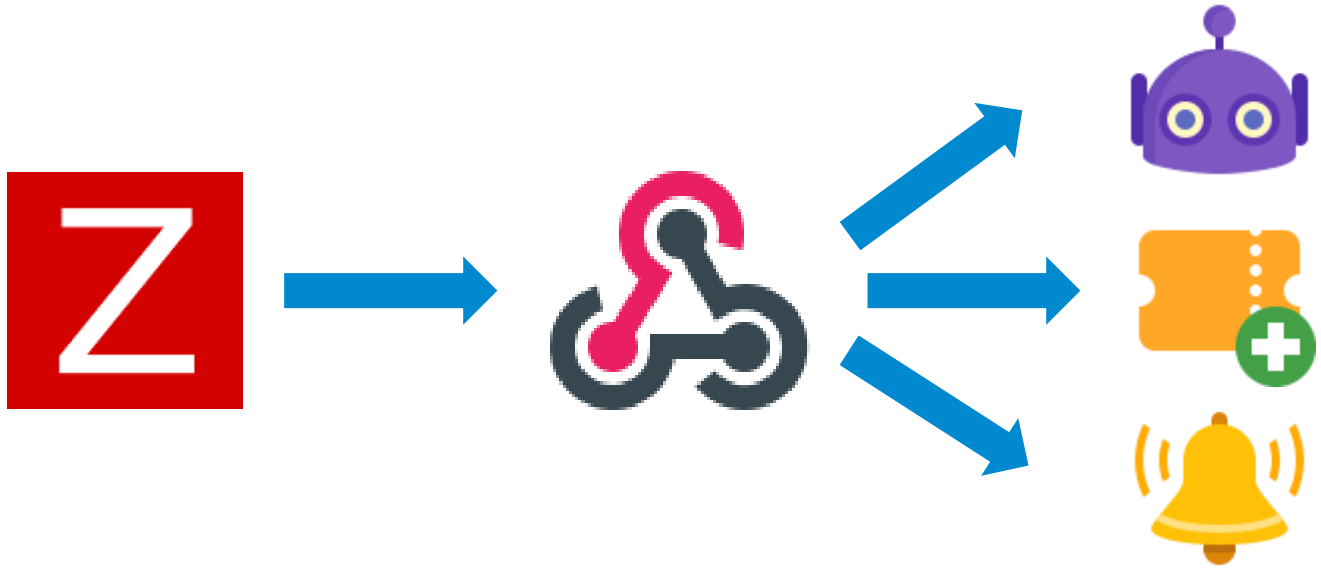
それ以外はソースからコンパイルが必要

次世代の Zabbix Agent

機能・特徴

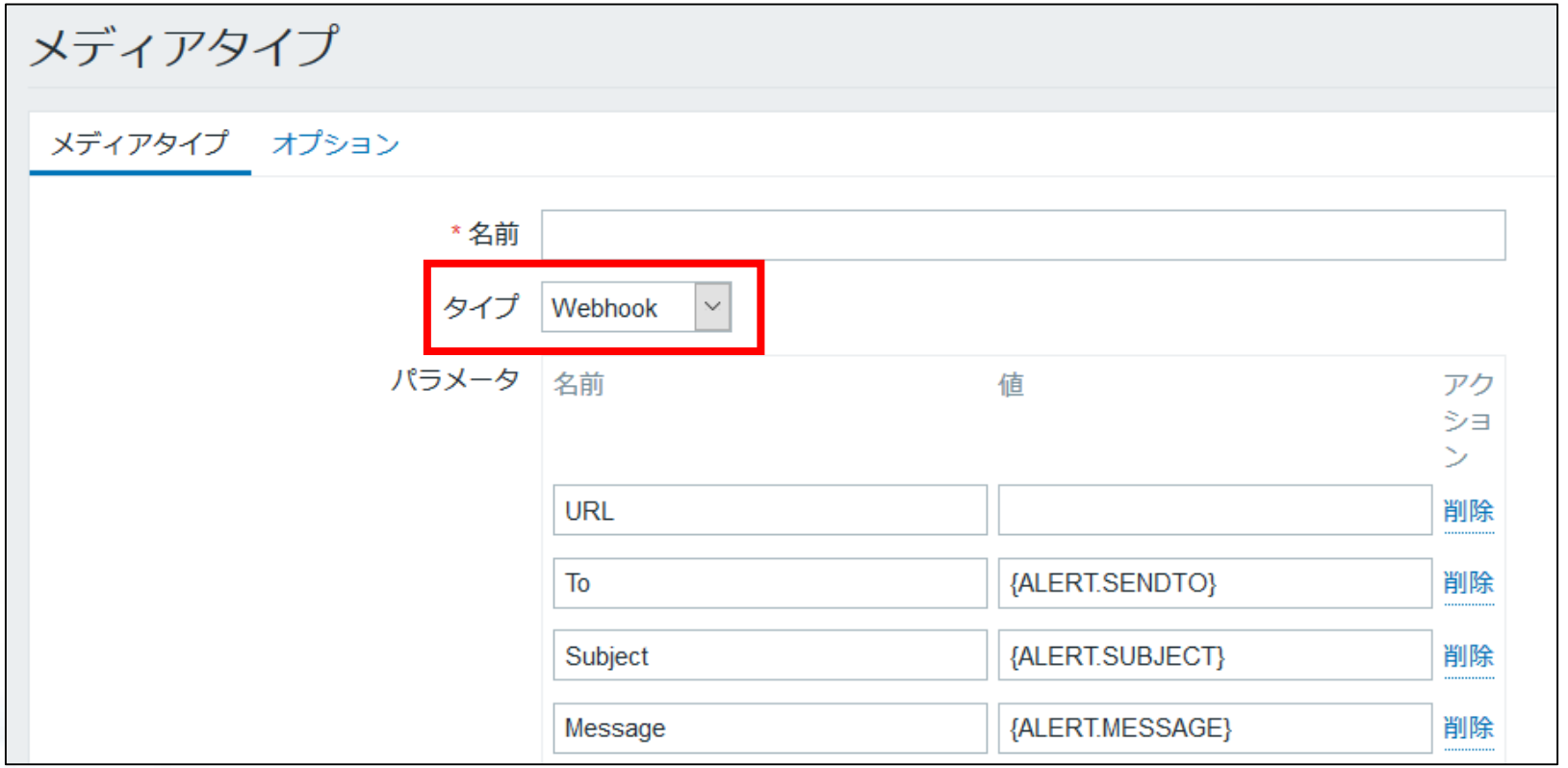


- ☑ 監視の高い並列実行性
- ☑ プラグインによる拡張容易性
- ☑ TCP 接続数の削減
- ☑ 現行の Zabbix Agent の機能は全て踏襲
- ☑ アクティブチェックの監視間隔カスタマイズが可能
- ☑現行の設定ファイルとほぼ互換性あり



Webhook によるアラート

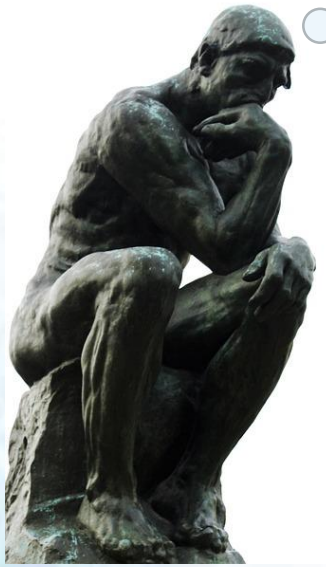
Webhook によるアラート



- ☑ メディアタイプに **Webhook** が追加
- ☑ 外部システムとの連携がより容易に

Webhook によるアラート

それってカスタムスクリプトと
どう違うの？



Webhook によるアラート

Frontend から JavaScript

JavaScript

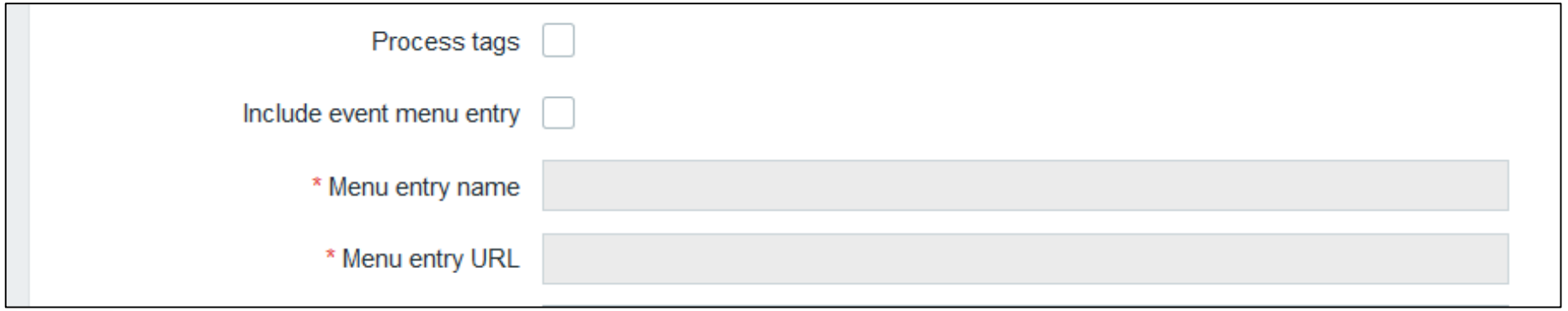
```
try {
  Zabbix.Log(4, 'Redmine webhook script value='+value);

  var result = {
    'tags': {
      'endpoint': 'Redmine'
    }
  },
  params = JSON.parse(value),
  req = new CurlHttpRequest(),
  issue = {},
  resp;

  req.AddHeader('Content-Type: application/json');
  req.AddHeader('X-Redmine-API-Key: '+params.api_key);
```

Webhook によるアラート

レスポンスの値を利用可能

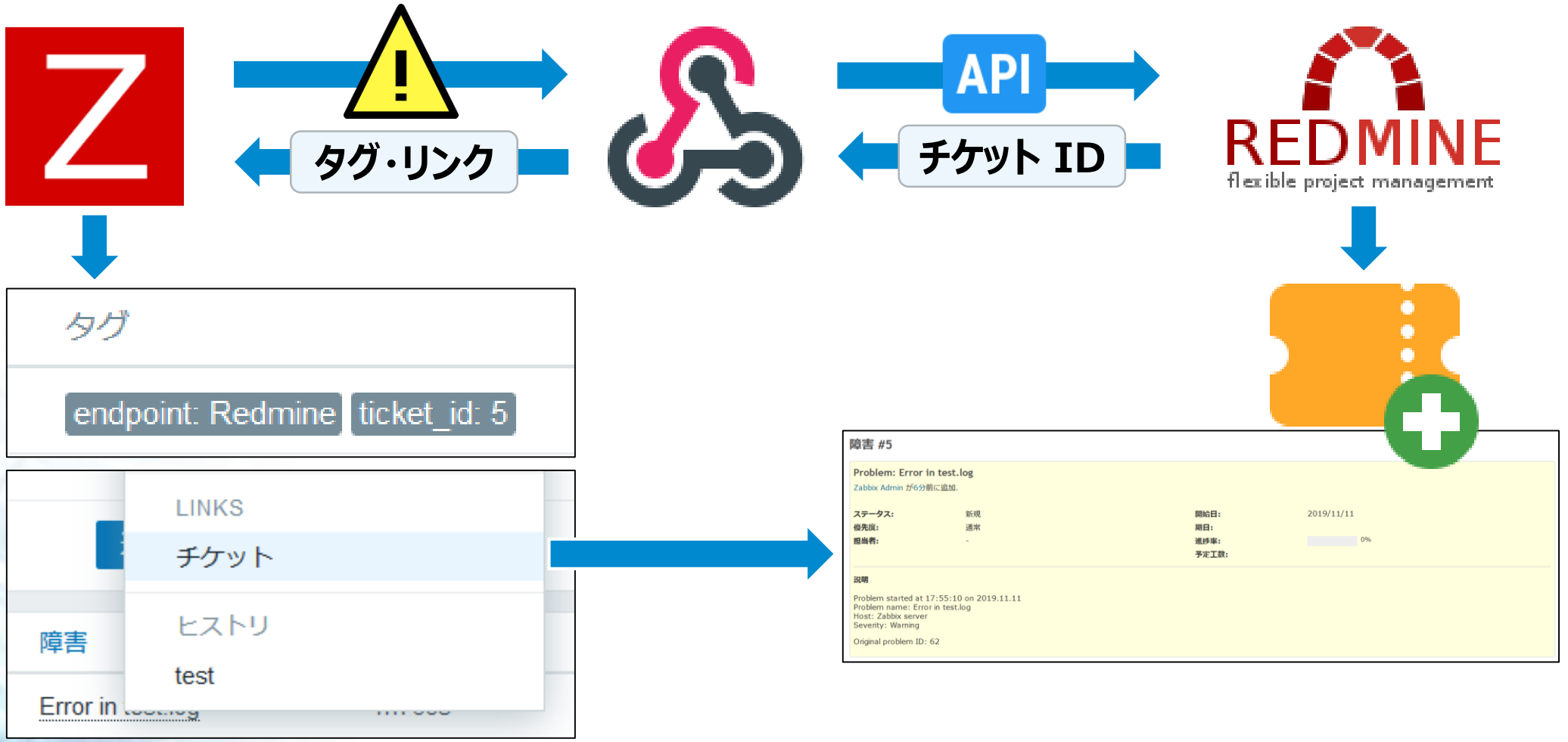


☑ Process tags
⇒ 特定の値を動的にイベントのタグに追加

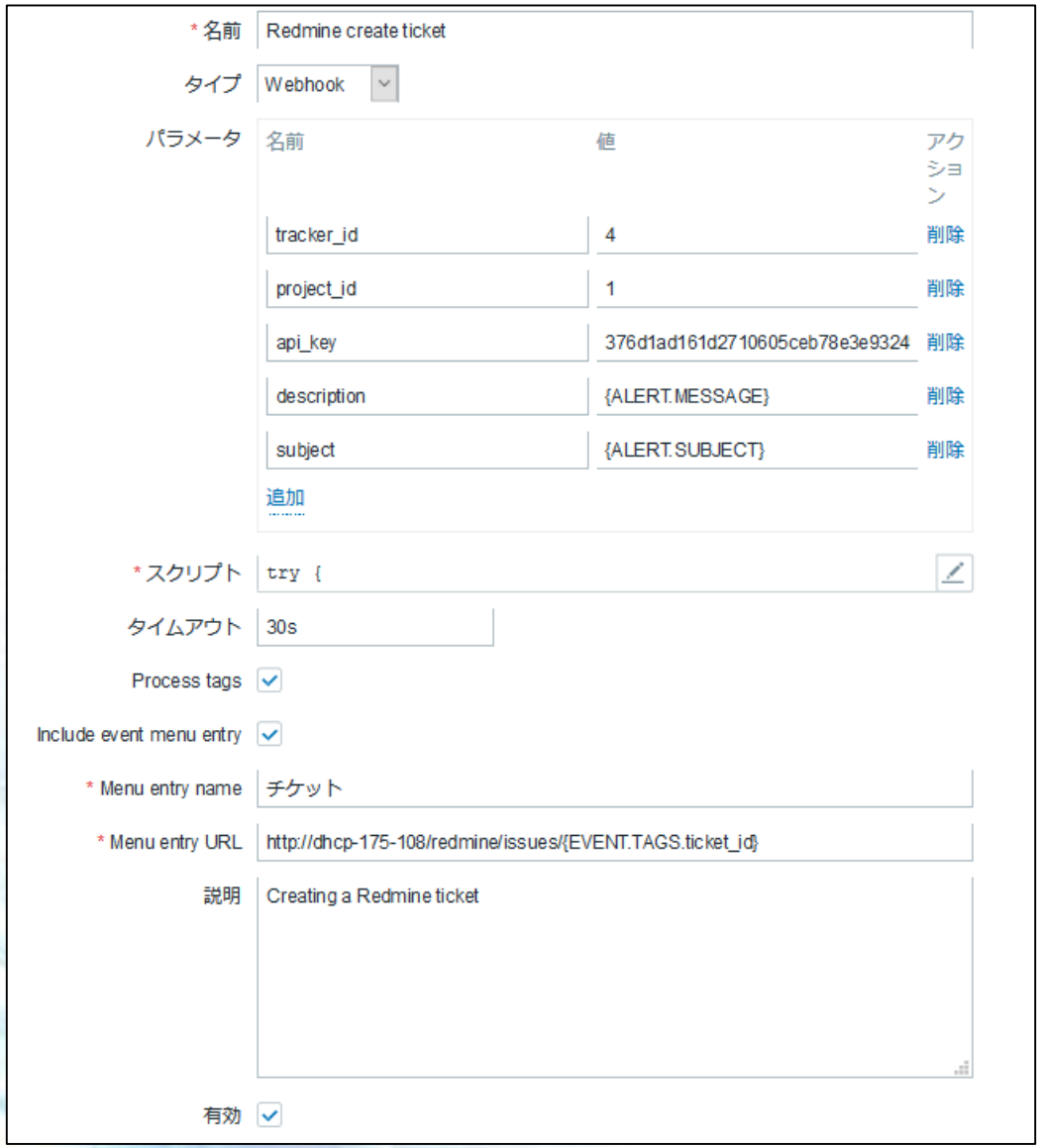
☑ Include event menu entry
⇒ イベントのメニューに動的に任意のリンクを追加

Webhook によるアラート

例：Redmine にチケットを登録



Webhook によるアラート



JavaScript

```
try {
  Zabbix.Log(4, 'Redmine webhook script value='+value);
  var result = {
    'tags': {
      'endpoint': 'Redmine'
    }
  },
  params = JSON.parse(value),
  req = new CurlHttpRequest(),
  issue = {},
  resp;

  req.AddHeader('Content-Type: application/json');
  req.AddHeader('X-Redmine-API-Key: '+params.api_key);
  issue.project_id = params.project_id;
  issue.subject = params.subject;
  issue.description = params.description;
  issue.tracker_id = params.tracker_id;
  resp = req.Post('http://dhcp-175-108/redmine/issues.json',
    JSON.stringify({"issue": issue})
  );

  if (req.Status() != 201) {
    throw 'Response code: '+req.Status();
  }

  resp = JSON.parse(resp);
  result.tags.ticket_id = resp.issue.id;
} catch (error) {
  Zabbix.Log(4, 'Redmine ticket creation failed json : '+JSON.stringify({"issue": issue}));
  Zabbix.Log(4, 'Redmine ticket creation failed : '+error);
  result = {};
}

return JSON.stringify(result);
```

64588文字入力可

TimescaleDB の公式サポート

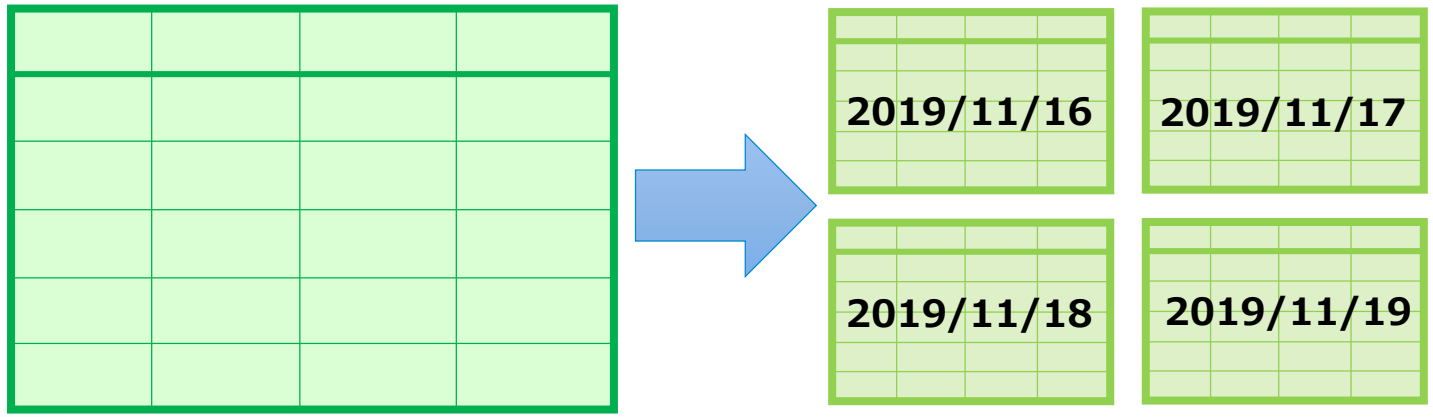
TimescaleDB とは？

時系列データに強い PostgreSQL

- PostgreSQL の EXTENSION
  - OSS
  - PostgreSQL との互換性
- パフォーマンス向上
  - データ削除が速い
  - クエリも速い
- 時系列データ特化
  - 時系列分析関数
  - 時系列データ管理機能

TimescaleDB のアーキテクチャ

テーブルを時間単位でパーティショニング



hypertable
(親テーブル)

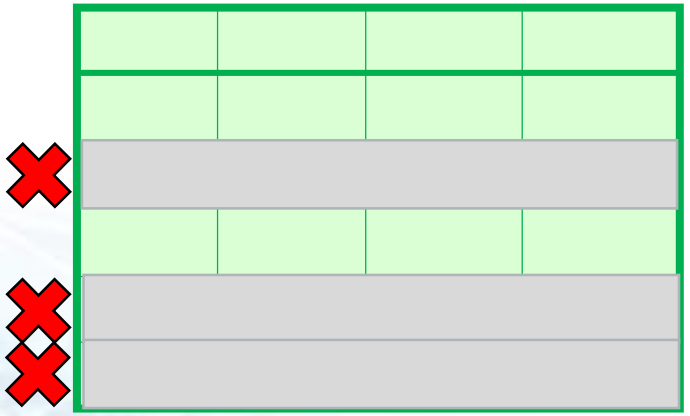
chunk
(子テーブル)

SELECT や INSERT などの親テーブルで実行した操作は、自動的に子テーブルに伝播

housekeeper の違い

PostgreSQL

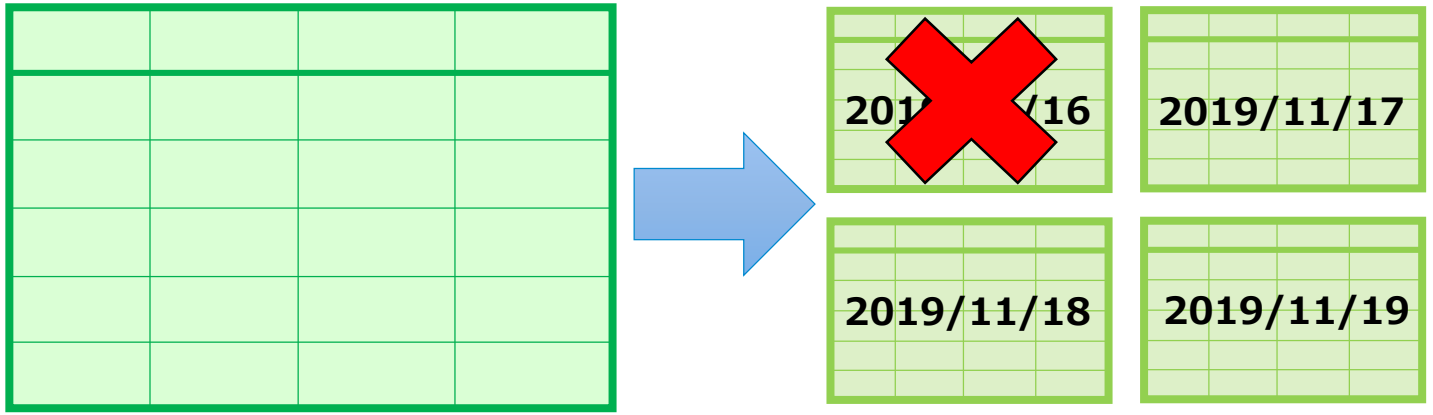
```
DELETE FROM history
WHERE
itemid = xxxx AND clock < yyyy
```



該当行を検索して削除
不要領域回収に要 VACUUM

TimescaleDB

```
SELECT
drop_chunks(zzzz, 'history')
```



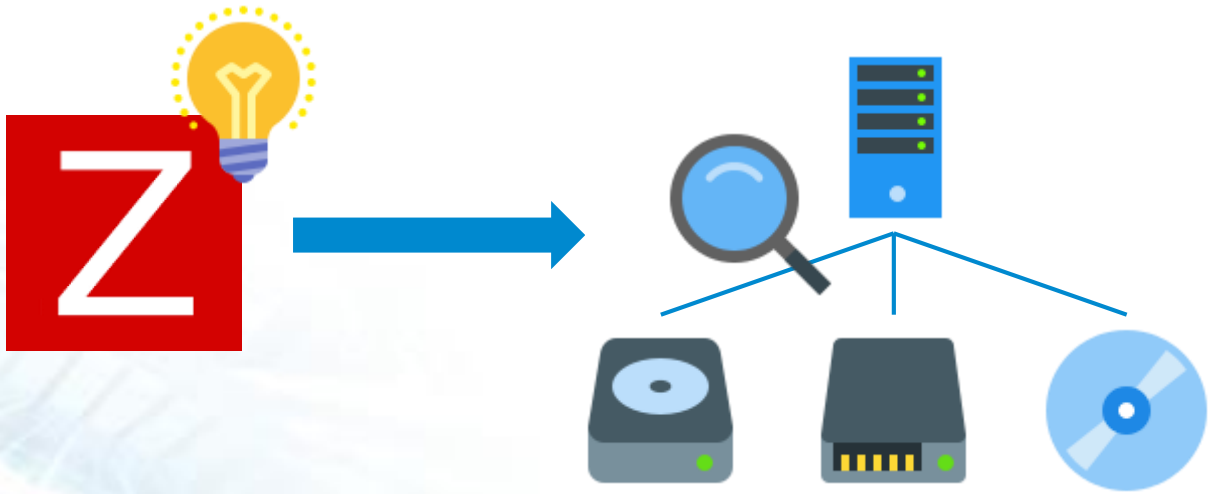
子テーブルごと削除
VACUUM 不要

LLD アイテムの追加

LLD アイテムの追加

ブロックデバイスのディスカバリ

vfs.dev.discovery



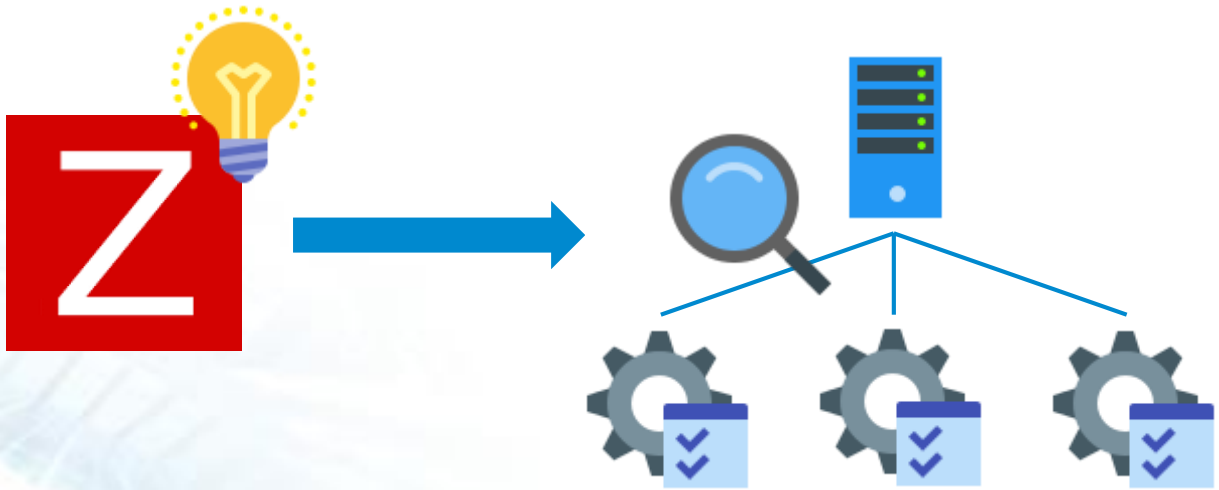
☑ Linux のみでサポート

```
[
  {
    “{#DEVNAME}“:”cdrom”,
    “{#DEVTYPE}”:“disk”
  },
  {
    “{#DEVNAME}”:“fd0”,
    “{#DEVTYPE}”:“disk”
  },
  {
    “{#DEVNAME}”:“sda”,
    “{#DEVTYPE}”:“disk”
  },
  {
    “{#DEVNAME}”:“sda1”,
    “{#DEVTYPE}”:“partition”
  },
  …
]
```

LLD アイテムの追加

systemd サービスのディスカバリ

systemd.unit.discovery



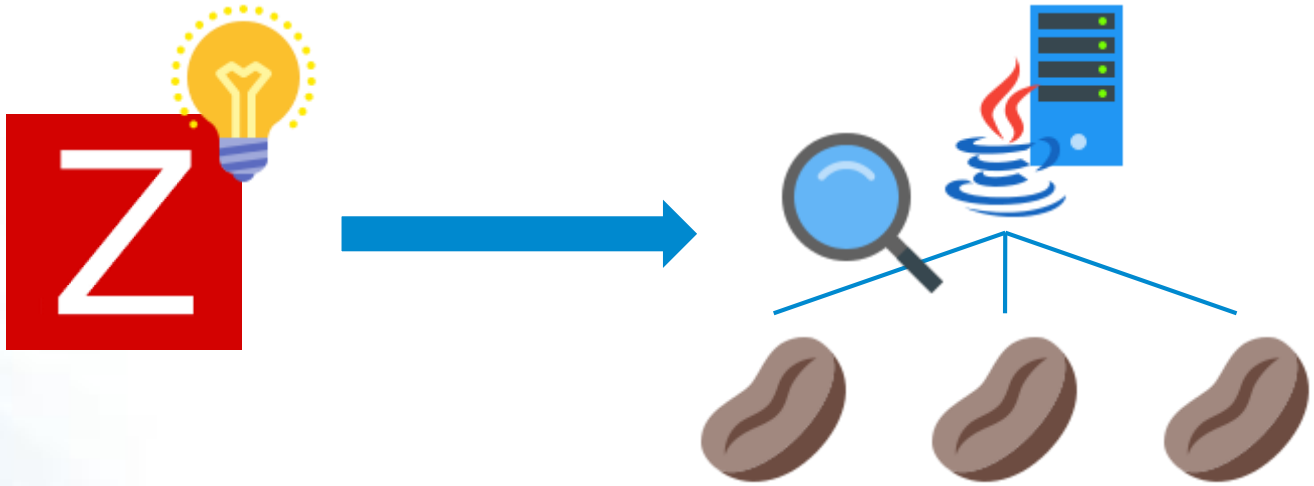
```
[
  {
    "{#UNIT.NAME}":"zabbix-server.service",
    "{#UNIT.DESCRIPTION}":"Zabbix Server",
    "{#UNIT.LOADSTATE}":"loaded",
    "{#UNIT.ACTIVESTATE}":"active",
    "{#UNIT.SUBSTATE}":"running",
    "{#UNIT.FOLLOWED}":"",
    "{#UNIT.PATH}":"/org/freedesktop/...",
    "{#UNIT.JOBID}":0,
    "{#UNIT.JOBTYPE}":"",
    "{#UNIT.JOBPATH}":"/"},
  },
  ...
]
```

☑ Zabbix Agent 2 かつ Linux のみでサポート

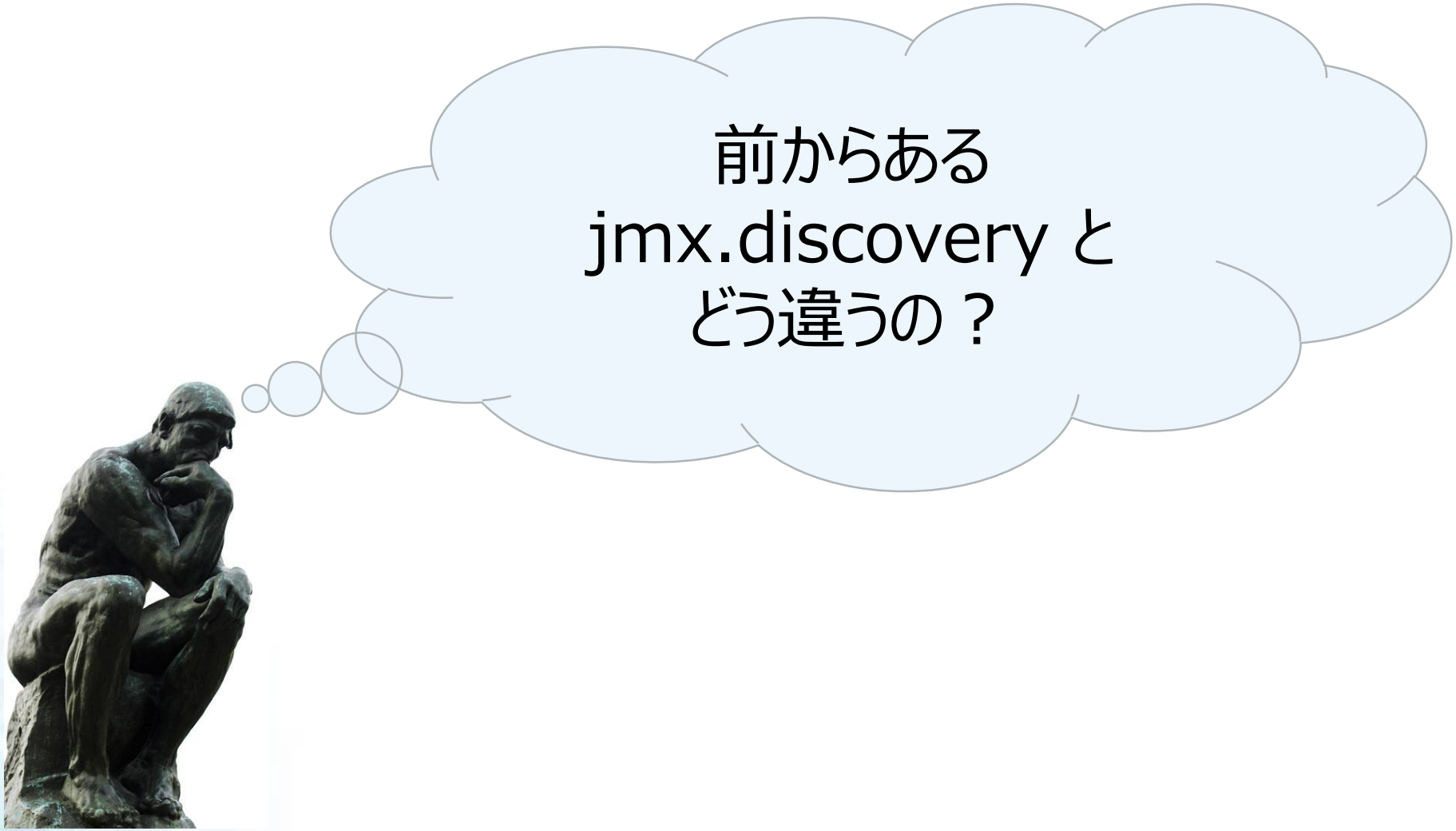
LLD アイテムの追加

JMX MBean のディスカバリ

jmx.get[<discovery mode>,<object name>]



LLD アイテムの追加



LLD アイテムの追加

jmx.discovery

ObjectName=“java.lang:type=MemoryPool,name=Metaspace,attr-1=hoge”

```
[
  {
    "{#JMXOBJ}":"java.lang:type=MemoryPool,name=Metaspace,attr-1=hoge",
    "{#JMXDOMAIN}":"java.lang",
    "{#JMXTYPE}":"MemoryPool",
    "{#JMXNAME}":"Metaspace“
  }
]
```

- ☑ MBean やその属性を **LLD マクロ付きの JSON** で返す
- ☑ マクロでサポートされない文字列が含まれる属性は無視
  - ☑ ハイフンや括弧、非アスキー文字など

LLD アイテムの追加

jmx.get

ObjectName=“java.lang:type=MemoryPool,name=Metaspace,attr-1=hoge”

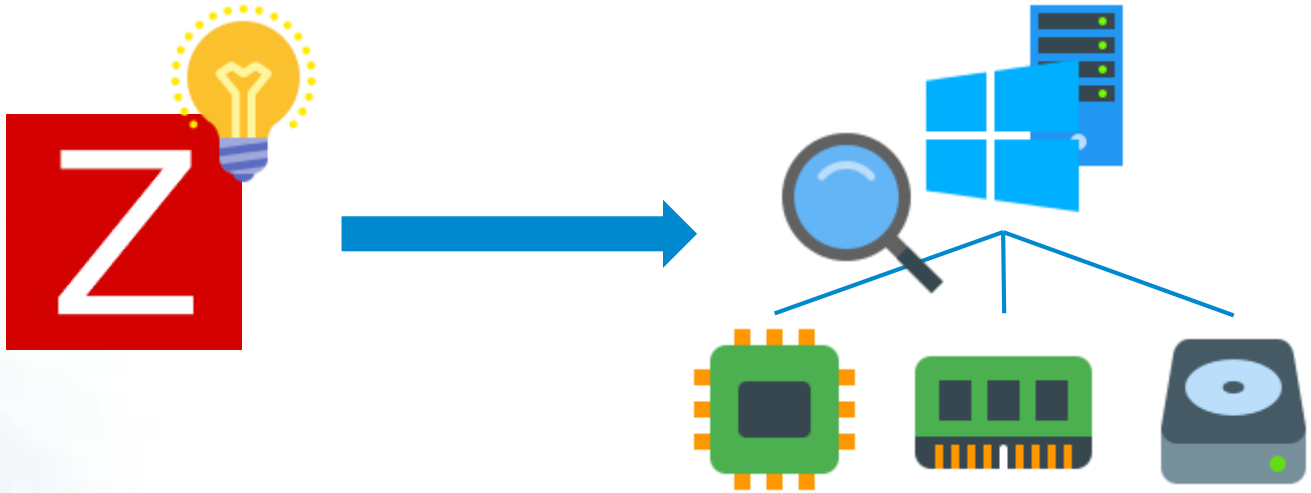
```
[
  {
    "object":"java.lang:type=MemoryPool,name=Metaspace,attr-1=hoge“
    "domain":"java.lang",
    "properties":{
      "name":"Metaspace",
      "type":"MemoryPool“,
      “attr-1":“hoge“}
  }
]
```

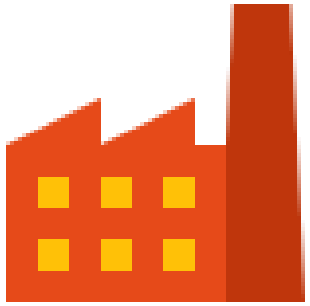
- ☑ MBean やその属性を **JSON** で返す
- ☑ キーに対するカスタム LLD マクロの定義が必要

LLD アイテムの追加

WMI のディスカバリ

```
wmi.getall[<namespace>,<query>]
```





保存前処理の新機能

- ☑ CSV to JSON
- ☑ JSON 内 / XML 内 / 正規表現使用時のエラーチェックでの失敗時のカスタマイズ
- ☑ LLD ルールで使用できる保存前処理の追加

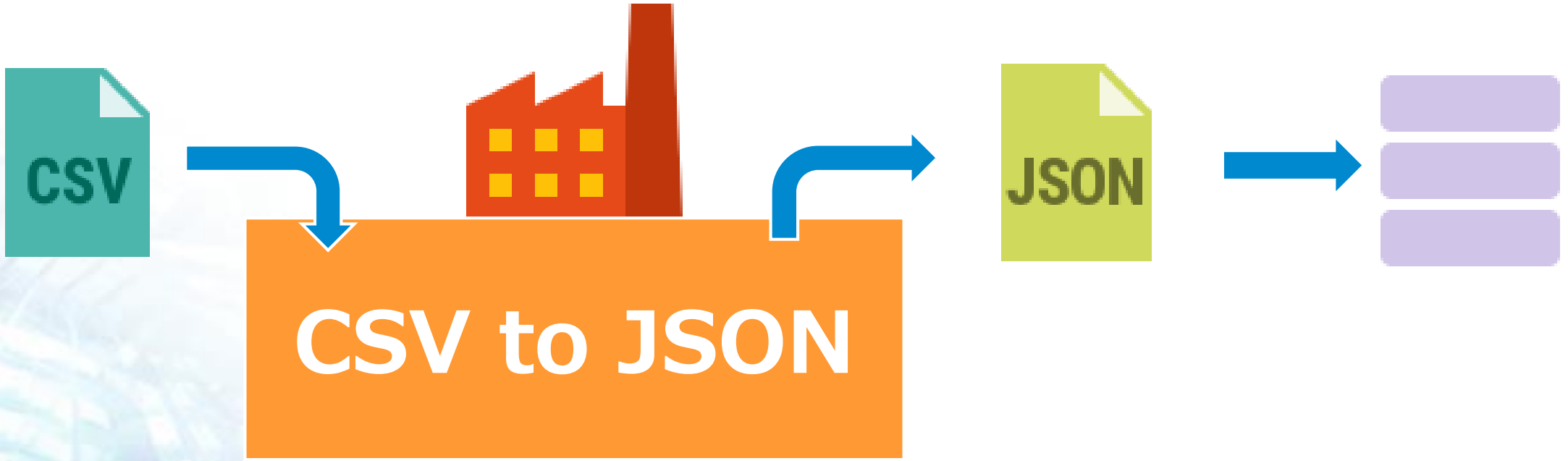
CSV to JSON

保存前処理の設定

名前　　パラメータ

1: CSV to JSON　,　"　With header row

追加



CSV to JSON

CSV to JSON のパラメータ

1. 最初の行をヘッダ行とする
2. デリミタの指定
3. 指定した文字を文字ではなくクォート記号として扱う

CSV to JSON

1 最初の行をヘッダ行とする

```
ID,Host,IP
1,host1,10.1.1.1
2,host2,10.1.1.2
…
```

☑ With header row

```
[
  {
    “ID”:”1”,
    “Host”:“host1”,
    “IP”:“10.1.1.1”
  },
  {
    “ID”:”2”,
    “Host”:“host2”,
    “IP”:“10.1.1.2”
  },
  …
]
```

☐ With header row

```
[
  {
    "1":"ID",
    "2":"Host",
    "3":"IP“
  },
  {
    "1":"1",
    "2":"host1",
    "3":"10.1.1.1“
  },
  …
]
```

CSV to JSON

2 デリミタの指定

```
ID,Host,IP
1,host1,10.1.1.1
2,host2,10.1.1.2
…
```


```
ID@Host@IP
1@host1@10.1.1.1
2@host2@10.1.1.2
…
```


```
[
  {
    “ID”:”1”,
    “Host”:“host1”,
    “IP”:“10.1.1.1”
  },
  {
    “ID”:”2”,
    “Host”:“host2”,
    “IP”:“10.1.1.2”
  },
  …
]
```

CSV to JSON

☑ ちなみに CSV の頭に Sep=<デリミタ> でも OK

```
Sep=@
ID@Host@IP
1@host1@10.1.1.1
2@host2@10.1.1.2
…
```


```
[
  {
    “ID”:”1”,
    “Host”:“host1”,
    “IP”:“10.1.1.1”
  },
  {
    “ID”:”2”,
    “Host”:“host2”,
    “IP”:“10.1.1.2”
  },
  …
]
```

CSV to JSON

☑ さらにパラメータと Sep= ではパラメータ優先

```
Sep=|
ID@Host@IP
1@host1@10.1.1.1
2@host2@10.1.1.2
…
```


```
[
  {
    “ID”:”1”,
    “Host”:“host1”,
    “IP”:“10.1.1.1”
  },
  {
    “ID”:”2”,
    “Host”:“host2”,
    “IP”:“10.1.1.2”
  },
  …
]
```

CSV to JSON

3 指定した文字を文字ではなくクォート記号として扱う

```
“ID”,Host,IP
1,host1,10.1.1.1
2,host2,10.1.1.2
…
```


```
[
  {
    “¥”ID¥””:”1”,
    “Host”:“host1”,
    “IP”:“10.1.1.1”
  },
  …
```

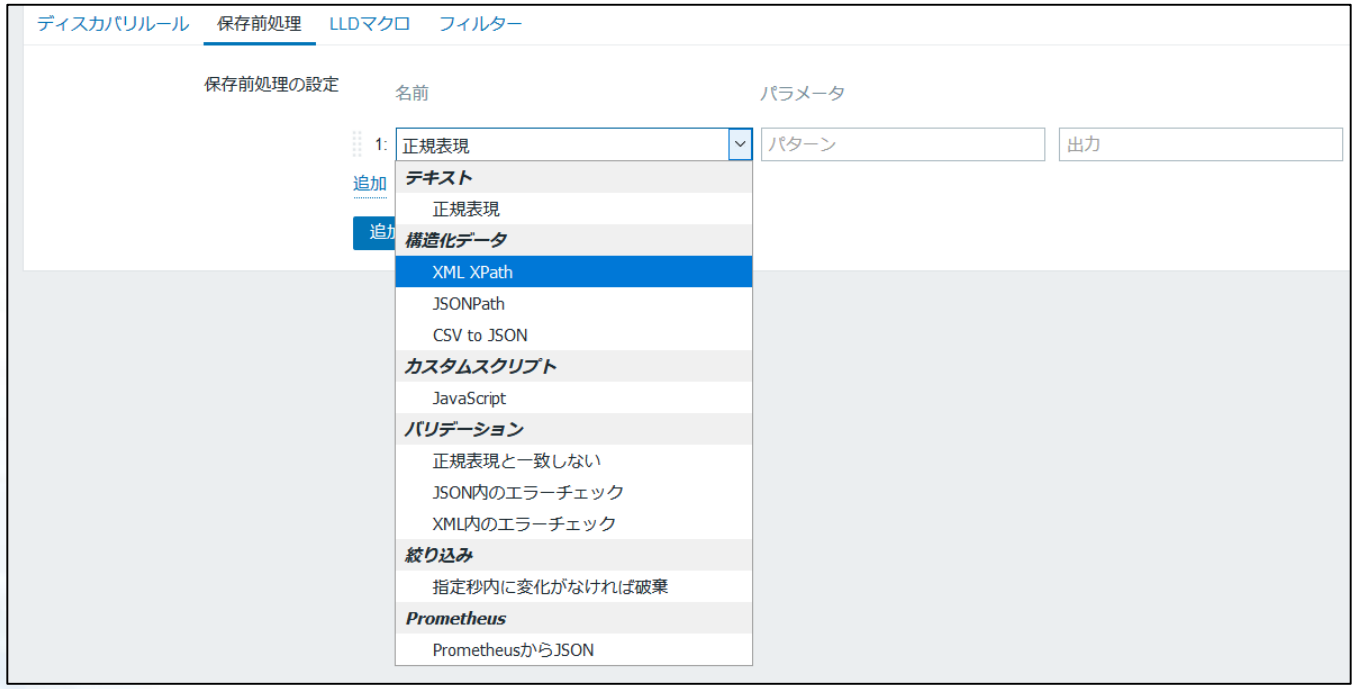

```
[
  {
    “ID”:”1”,
    “Host”:“host1”,
    “IP”:“10.1.1.1”
  },
  …
```

失敗時のカスタマイズ

☑ JSON 内 / XML 内 / 正規表現使用時の
エラーチェックで失敗時のカスタマイズが使用可能

LLD ルールでの保存前処理

- ☑ LLD ルールで使用できる保存前処理に以下が追加
  - ☑ XML Xpath
  - ☑ CSV to JSON
  - ☑ XML 内でのエラーチェック

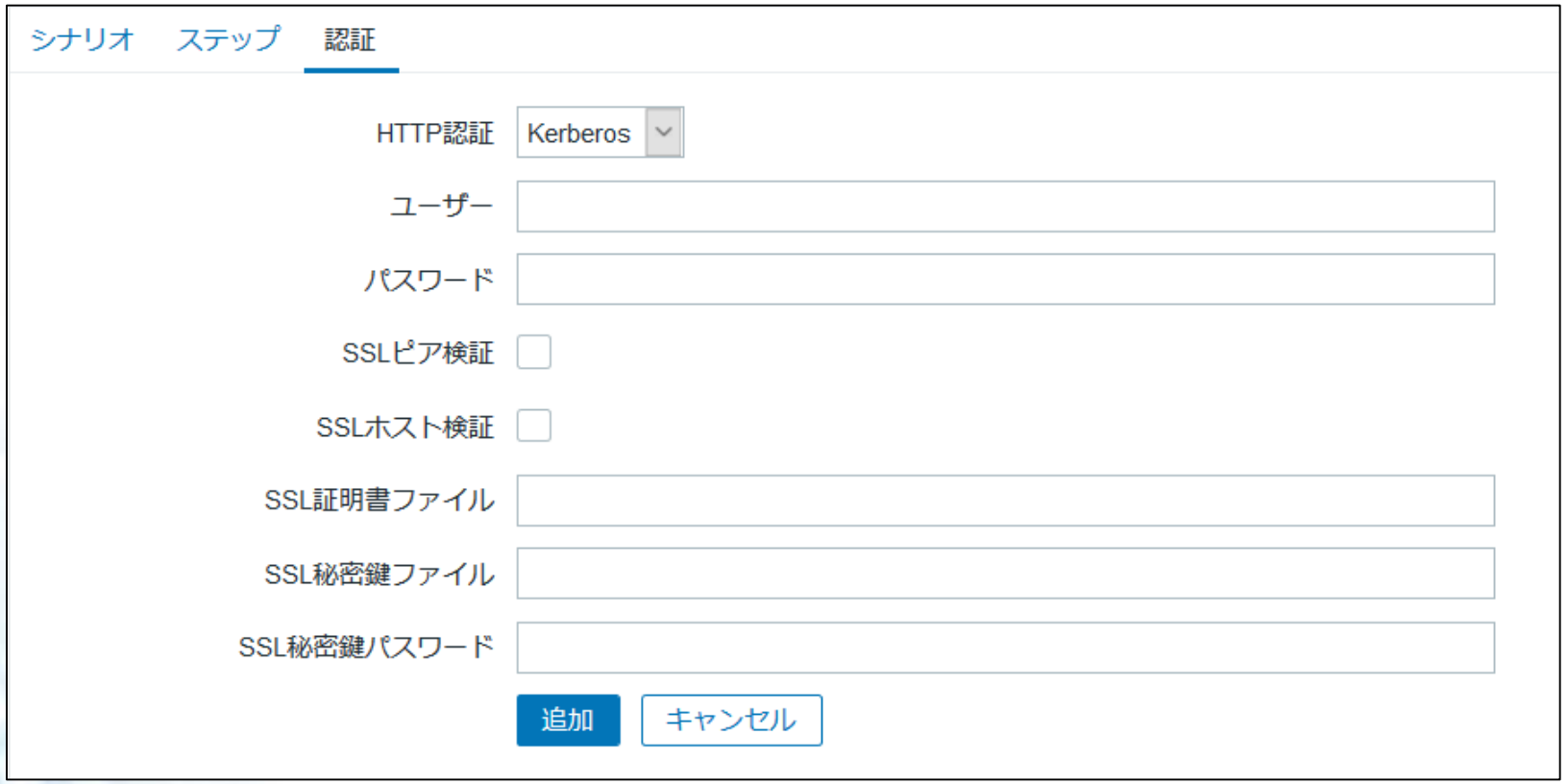


セキュリティの新機能

- ☑ Web 監視および HTTP アイテムに Kerberos 認証サポートが追加
- ☑ 自動登録の通信に PSK ベースの暗号化サポートが追加

Kerberos 認証サポート

☑ Web 監視と HTTP エージェントの認証方法に Kerberos 認証が追加



セキュアな自動登録

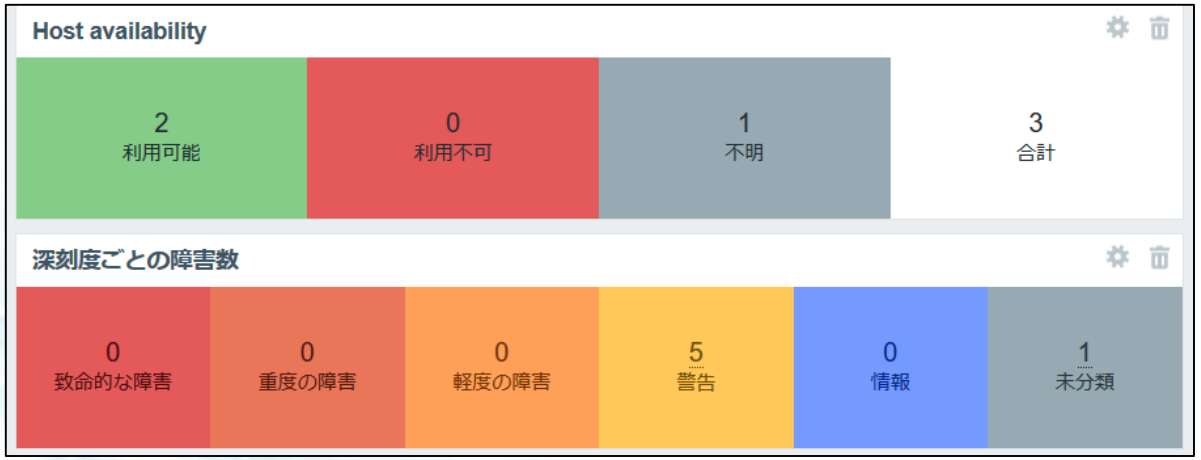
☑ エージェントの自動登録で PSK ベースの通信が可能に

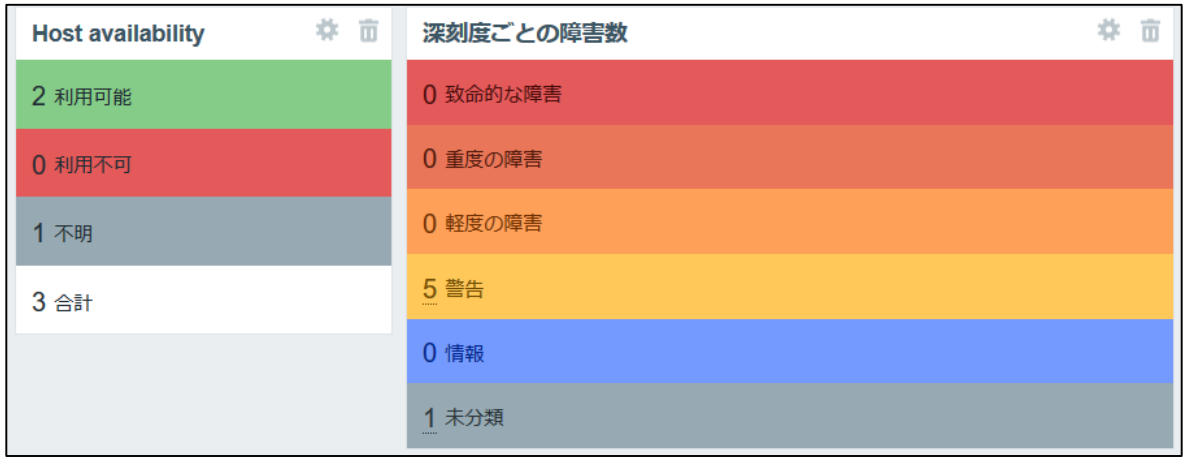
Frontend の新機能

- ☑ Host availability の追加
- ☑ ウィジェットのヘッダ非表示機能
- ☑ bar グラフの追加
- ☑ グラフでの集約関数
- ☑ Operational data 表示
- ☑ アイテム/トリガーの説明表示

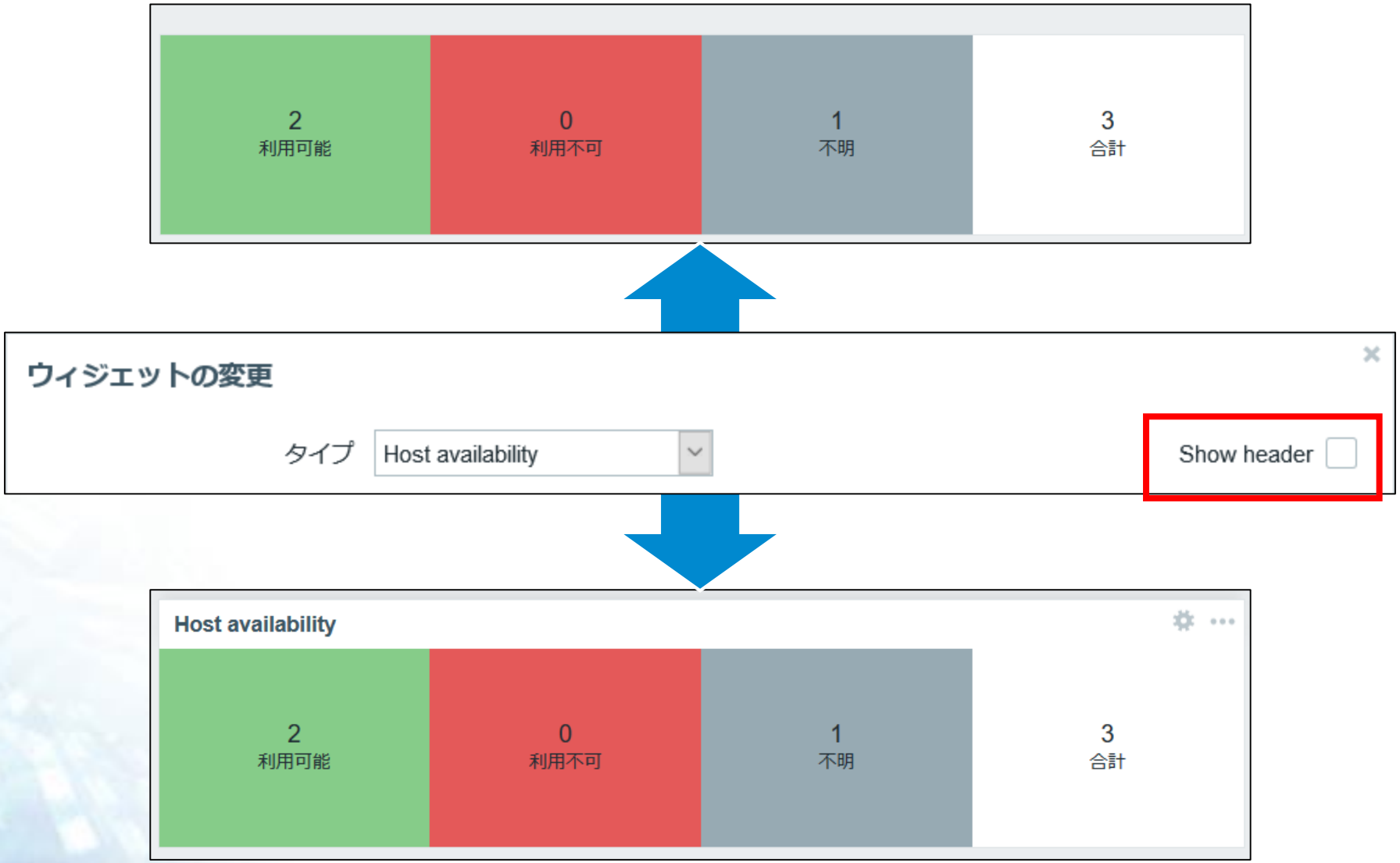
Host availability の追加

☑ Host Availability ウィジェットが追加
☑ 深刻度ごとの障害数の見た目も変更

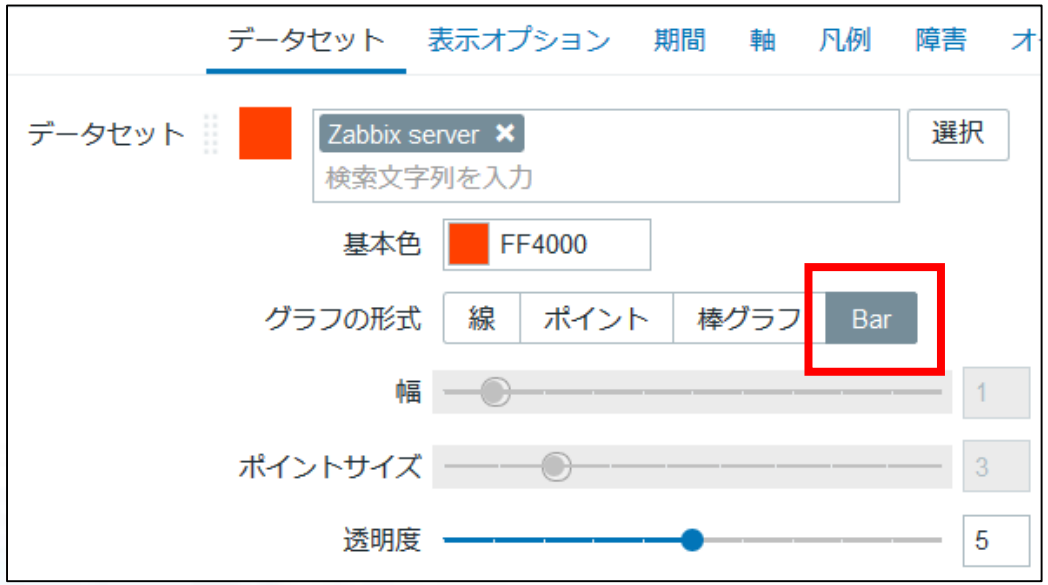


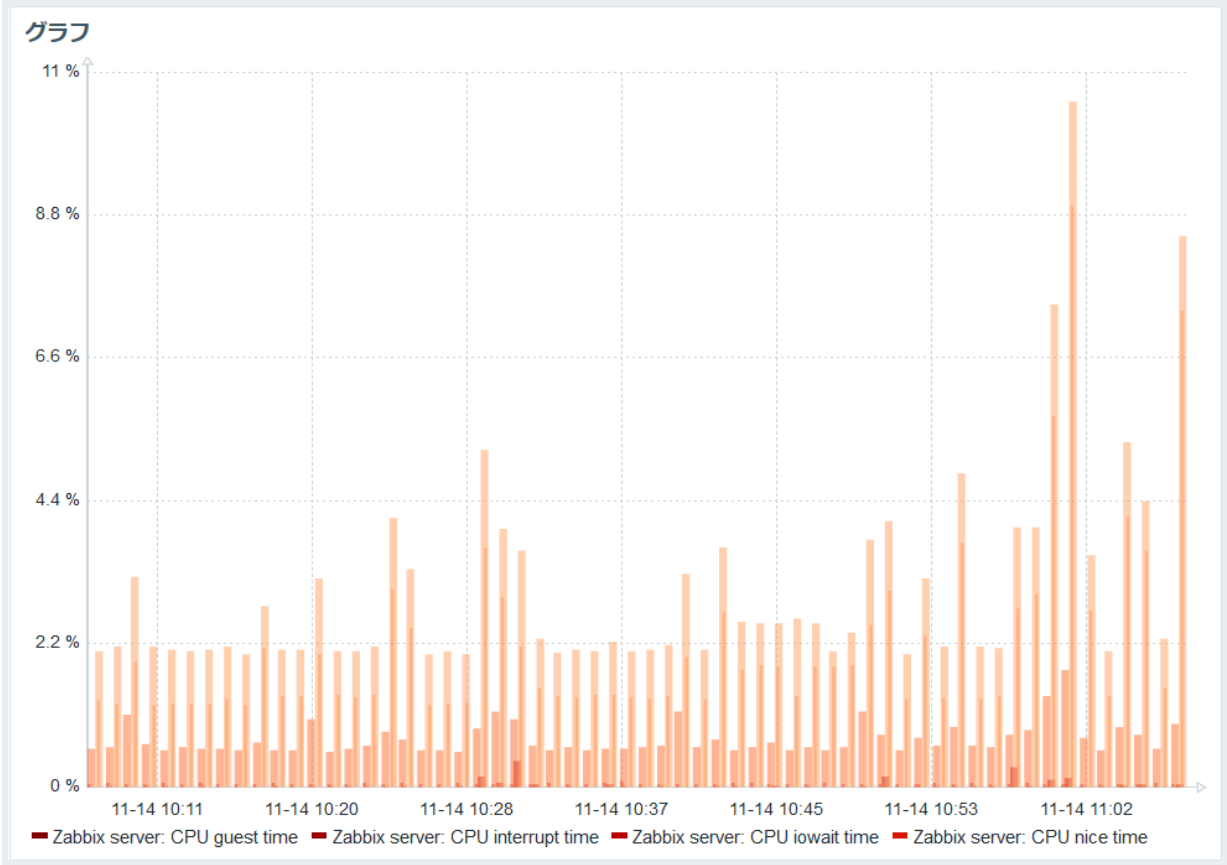


ウィジェットのヘッダ非表示



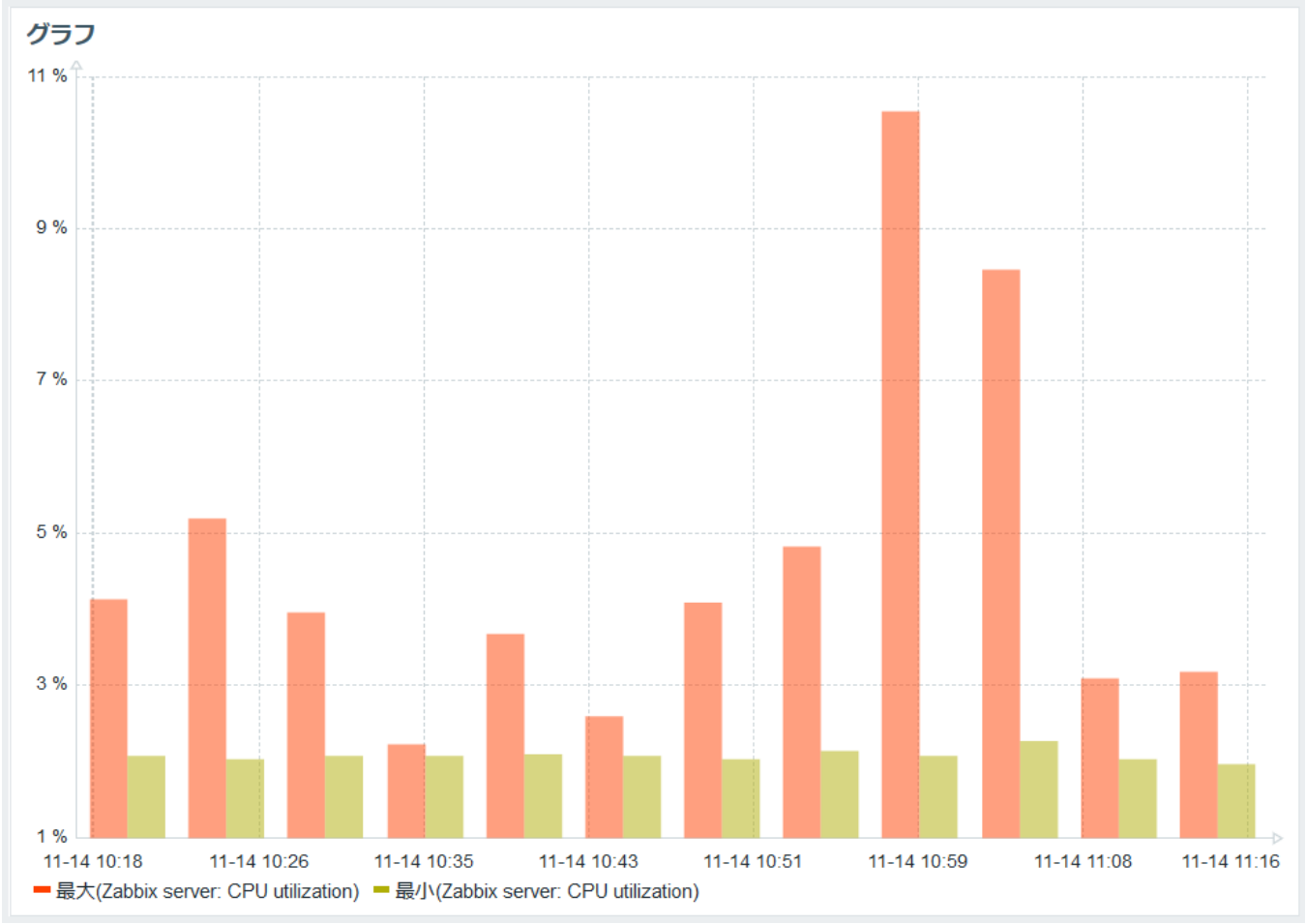
Bar グラフの追加





グラフでの集約関数

- ☑ Aggregation function
  - ☑ なし
  - ☑ 最小
  - ☑ 最大
  - ☑ 平均
  - ☑ count
  - ☑ sum
  - ☑ 第 1
  - ☑ 最新値



Operational data 表示

トリガー設定

トリガー　タグ　依存関係

* 名前　High CPU utilization (over {$CPU.UTIL.CRIT}% for 5m)

Operational data　Current utilization: {ITEM.LASTVALUE1}

障害画面

Show operational data　なし　Separately　With problem name

時間	深刻度	復旧時刻	ステータス	情報	ホスト	障害	Operational data	継続期間
11:35:13	警告		障害		Zabbix server	High CPU utilization (over 90% for 5m)	Current utilization: 100 %	5m 8s

Show operational data　なし　Separately　With problem name

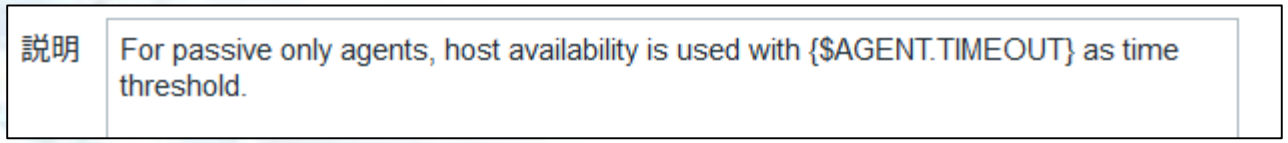
時間	深刻度	復旧時刻	ステータス	情報	ホスト	障害	継続期間
11:35:13	警告		障害		Zabbix server	High CPU utilization (over 90% for 5m) (Current utilization: 100 %)	5m 56s

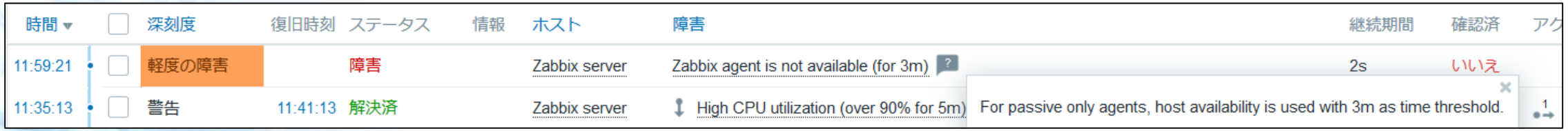
アイテム/トリガーの説明表示

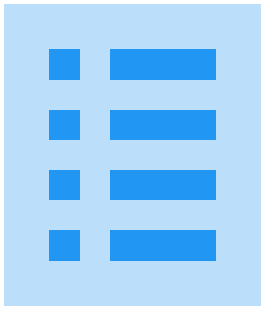
☑ アイテム/トリガーの説明を tooltop で表示

アイテム

トリガー







その他の新機能

その他の新機能

- ☑ db.odbc.get アイテムの追加
- ☑ ホスト一覧画面などでプロキシの列が分離
- ☑ トリガー条件式のフォントが等幅に変更
- ☑ DNS 名での自動登録
- ☑ メディアタイプのエクスポート/インポート
- ☑ エクスポートファイルの構造の改善
- ☑ items テーブルから一部のフィールドを item_rtdata テーブルに分離