

ZABBIX 2021 Conference JAPAN



コンテナでの監視機能の実装と problemテーブル肥大化問題の事例紹介

2021年11月18日

エヌ・ティ・ティ・コミュニケーションズ株式会社

名倉 堂心 (なくら たかみ)



名倉 堂心 (なくら たかみ)

【経歴】

- 2020年4月 新卒でNTTコムソリューションズ入社（入社2年目）
- 2020年9月～ Zabbixの構築・運用業務に従事
- 2021年7月 NTTコミュニケーションズ合併後もZabbixに関わる

【スポーツ歴】

サッカー歴9年、空手歴9年、ラグビー歴8年目（継続中）

NTT コミュニケーションズ株式会社

(<https://www.ntt.com/>)

- ・ 2008年より、Zabbix社と提携したZabbix関連事業を開始
- ・ ZABICOMソリューションの導入/運用/製品供給などのサービスを提供



<https://www.zabicom.com>

1. コンテナでの監視機能の実装について
2. problemテーブル肥大化に関する事例と対応について
3. ちょっとだけ宣伝
4. まとめ

コンテナでの監視機能の実装について

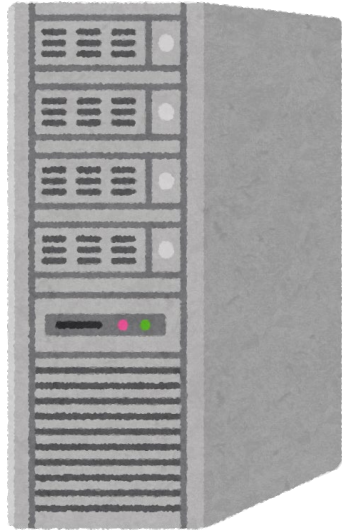
どんな要件があるのか？

要件

- ① ネットワーク・システムの監視をしたい
- ② オンプレ、仮想環境の監視にZabbixを導入したい

**監視システムとして、Zabbixを
物理サーバまたは仮想サーバで導入するケースがほとんど**

Zabbixの導入時に用いるプラットフォームは？



物理マシン



仮想マシン



コンテナ

今回お話するケースでは…

検証環境でOKが出たものを、本番環境に適用したい

という要件も…



検証環境から本番環境へ移植・適用する作業が必要

コンテナ使うのに
なんでひとまとめにしてんの？

オンプレでの設計と
変わんかない？

コンテナ使う意味ww

A.ごもっともです。。。

仮にマイクロサービス化した場合



で、保守・運用どうするの？

たしかに(;'▽')
実績もないしどうしよう。。。



1台に複数のマイクロサービスコンテナでZabbixを構築する場合との比較

メリット

- ✓ オンプレとほぼ同様の構築手順
- ✓ オンプレ構築と変わらない保守性、運用性



自分たちが保守・運用をするにあたっても確実な手段

つまるどころ…



物理マシン



仮想マシン

運用実績の多さ

いいとこどり◎



コンテナ

持ち運びのしやすさ

- ✓ 検証環境から本番環境への適用がスムーズ
- ✓ 自分たちが保守・運用するにあたっても安心



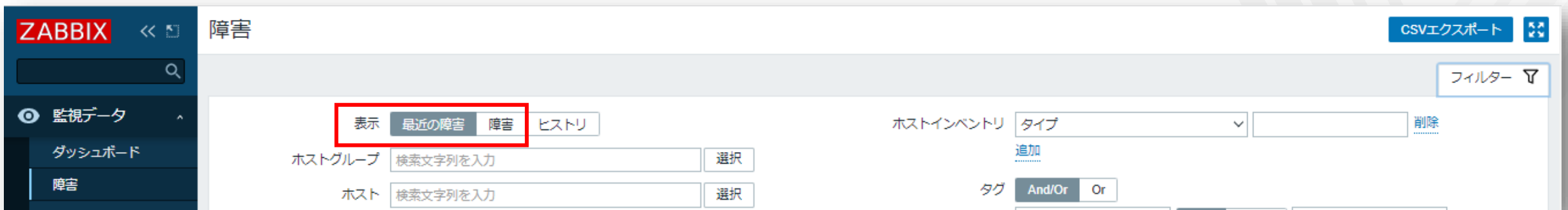
All in Oneコンテナが最適だった

problemテーブル肥大化に関する 事例と対応について

problemテーブルとは？

データベース内の「障害中」のイベントデータを格納する場所（テーブル）

「監視データ」→「障害」画面の**最近の障害**または**障害**が参照するテーブル



The screenshot shows the Zabbix web interface for the '障害' (Incidents) section. The left sidebar contains navigation links for '監視データ' (Monitoring Data), 'ダッシュボード' (Dashboard), and '障害' (Incidents). The main content area has a '表示' (Display) tab selected, with sub-tabs for '最近の障害' (Recent Incidents), '障害' (Incidents), and '履歴' (History). The '最近の障害' sub-tab is highlighted with a red box. Below the tabs are search filters for 'ホストグループ' (Host Groups) and 'ホスト' (Hosts), each with a search input field and a '選択' (Select) button. On the right, there are filters for 'ホストインベントリ' (Host Inventory) and 'タグ' (Tags), along with a 'CSVエクスポート' (Export to CSV) button and a 'フィルター' (Filter) button.

※参考※

https://assets.zabbix.com/img/zabconf2020_jp/presentations/02_2zabconf2020.pdf

「絶賛障害中のイベントが復旧したよ！」が正常イベントの定義



**障害中のデータは復旧しない限り
データの保存期間を過ぎてもテーブルに残り続ける**

「problemテーブル肥大化問題」ってナニ？

障害イベントが蓄積することで

ZabbixのWeb画面の表示が遅くなる、あるいは、表示できなくなる問題

The screenshot shows the Zabbix Global view dashboard. The left sidebar contains navigation menus for Monitoring Data, Dashboards, Incidents, Hosts, Summary, Latest Data, Screens, Maps, Discovery, Services, Inventory, Reports, Settings, Management, Support, Share, Help, and User Settings. The main content area is titled 'Global view' and includes a search bar and a 'すべてのダッシュボード / Global view' link. Below this is a 'システム情報' (System Information) section with a table of parameters. To the right is a clock widget. The '障害' (Incidents) section features a table with columns for time, host, incident name, duration, status, and actions. A summary bar shows 3 usable, 0 unusable, 0 unknown, and 3 total incidents. A red box highlights a '深刻度ごとの障害数' (Number of incidents by severity level) bar chart with the following data:

| Severity Level | Count |
|--------------------|-------|
| 致命的な障害 (Critical) | 0 |
| 重度の障害 (Major) | 0 |
| 軽度の障害 (Minor) | 0 |
| 警告 (Warning) | 3 |
| 情報 (Information) | 0 |
| 未分類 (Unclassified) | 0 |

復旧しない障害イベントが蓄積すること

トリガー

すべてのホスト / HostA 有効 ZBX SNMP JMX IPMI アプリケーション アイテム 20 トリガー 19 グラフ ティ...

トリガー タグ 依存関係

*名前 ろく検知

運用データ

深刻度 未分類 情報 警告 軽度の障害 重度の障害 致命的な障害

*条件式 {HostA:log[/hoge/moga.log].regexp(ERROR)}=1 追加

条件式ビルダー

正常イベントの生成 条件式 復旧条件式 なし

障害イベント生成モード 単一 複数

正常時のイベントクローズ すべての障害 タグの値が一致したすべての障害

手動クローズを許可

- ✓ イベント生成モード：複数
- ✓ 手動クローズを行わない運用

原因になりやすい監視

- ✓ ログ監視
- ✓ SNMPTrap監視

problemテーブルを肥大化させないために

- ✓ 復旧イベントを生成する
- ✓ 定期的に手動クローズを行う



The screenshot shows the Zabbix web interface for configuring a trigger. The left sidebar contains navigation options like '監視データ', 'インベントリ', 'レポート', '設定', and '管理'. The main content area is titled 'トリガー' (Trigger) and shows the configuration for a trigger named 'ろく検知'. The configuration includes a name field, a severity level set to '警告' (Warning), and a condition for the trigger: `{HostA:log[/hoge/moga.log,ERROR].regexp(ERROR)}=1`. Below this, the '条件式ビルダー' (Condition Builder) section is highlighted with a red box, showing the '正常イベントの生成' (Generate normal event) section with the '復旧条件式' (Recovery condition) tab selected. The recovery condition is set to `{HostA:log[/hoge/moga.log,ERROR].nodata(30)}=1`. Another red box highlights the '手動クローズを許可' (Allow manual close) checkbox, which is checked. The URL field at the bottom is empty.

復旧条件式の設定方法（regex関数を使用する際）

監視するログファイル

監視するログファイルに存在する正規表現

障害条件式 {HostA:log[/var/log/messages,"ERROR"].regex("ERROR",30)} = 1

アイテムキー

regex関数：最新の値に第1引数の正規表現が、直近の第2引数に指定した期間内に存在するか判定する関数

重要

復旧条件式 {HostA:log[/var/log/messages,"ERROR"].nodata(30)} = 1

アイテム：監視データ収集の定義

トリガー：アイテムの障害判定の定義

nodata関数：特定の期間内にアイテムを取得してないかどうかを判定する関数

※引用元※

https://assets.zabbix.com/img/zabconf2020_jp/presentations/02_2zabconf2020.pdf

regexp関数の第2引数がなぜ必要になるのか

障害条件式 {HostA:log[/var/log/messages,"ERROR"].regexp("ERROR",**30**)} = 1

復旧条件式 {HostA:log[/var/log/messages,"ERROR"].nodata(30)} = 1

復旧イベントが生成される条件は…

障害条件式が成立しない、かつ、**復旧条件式が成立する**場合のみ

最新値の検索期間を制限する必要がある

problemテーブル内にデータが200万件以上溜まり
Web画面にアクセスできなくなる

DBから直接データの削除を試みるも事態は好転せず...

DBの作り直し

```
MySQL [zbx0003je]> select count(*) from problem where r_clock=0 ;
```

```
+-----+  
| count(*) |  
+-----+  
| 2441611 |  
+-----+  
1 row in set (5.844 sec)
```



一定期間データを取得しない場合に、自動で復旧させる条件式を設定

nodata関数を使用

現在は正常に運用ができています◎

自動復旧条件式の設定のしかた注意！

ログ監視で、第2引数を指定しないで
アイテムを取得することはよくありますよね？



アイテム: `log[/var/log/messages]`

トリガー①: `{HostA:log[/var/log/messages].regexp("ERROR")}` = 1

トリガー②: `{HostA:log[/var/log/messages].regexp("CRITICAL")}` = 1

同じアイテムに複数のトリガーを設定する場合

トリガー①

障害条件式 {HostA:log[/var/log/messages].regexp("ERROR",30)}=1

復旧条件式 {HostA:log[/var/log/messages].nodata(30)} = 1

トリガー②

障害条件式 {HostA:log[/var/log/messages].regexp("CRITICAL",30)}=1

復旧条件式 {HostA:log[/var/log/messages].nodata(30)} = 1

どんなことが起こるか？

2021/11/18 **12:00:00** → ERROR → ERROR の障害を検知

2021/11/18 **12:00:01** → ERROR → ERROR の障害を検知

2021/11/18 **12:00:02** → CRITICAL → CRITICALの障害を検知



ERRORの障害も検知

原因は何??

トリガー①

障害条件式 {HostA:log[/var/log/messages].regexp("ERROR",30)} = 1

トリガー②

障害条件式 {HostA:log[/var/log/messages].regexp("CRITICAL",30)} = 1

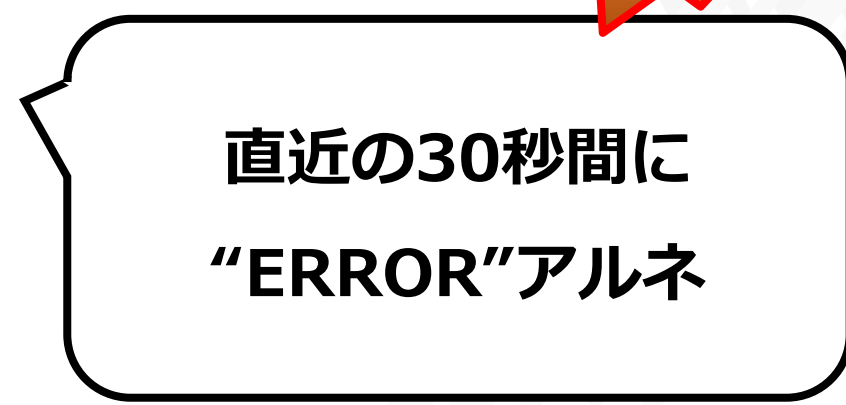
2021/11/05 12:00:00 "ERROR"

2021/11/05 12:00:01 "ERROR"

2021/11/05 12:00:02 "CRITICAL"



障害!



直近の30秒間に
"ERROR"アルネ

このような誤検知が起きないようにするために



The screenshot shows the Zabbix web interface for configuring an item. The left sidebar contains navigation menus for '監視データ', 'インベントリ', 'レポート', and '設定'. The main content area is titled 'アイテム' and shows the configuration for an item named 'エラーログ監視' with the key 'log[/hoge/moga.log,ERROR]'. The key field is highlighted with a red underline. A blue callout box points to the key field with the text 'アイテムキーの第2引数を指定！'.

| | |
|------|---------------------------|
| * 名前 | エラーログ監視 【/hoge/moga.log】 |
| タイプ | Zabbixエージェント(アクティブ) ▼ |
| * キー | log[/hoge/moga.log,ERROR] |
| データ型 | ログ ▼ |

データ収集の段階で、取り込むデータを絞り込む！

忘れてはいけないのは…

目的はproblemテーブルを肥大化させないこと

トリガー

すべてのホスト / HostA 有効 ZBX SNMP JMX IPMI アプリケーション アイテム 20 トリガー 19 グラフ ディスカ

トリガー タグ 依存関係

*名前 ろぐ検知

運用データ

深刻度 未分類 情報 警告 軽度の障害 重度の障害 致命的な障害

*障害の条件式 {HostA:log[/hoge/moga.log,ERROR].regexp(ERROR)}=1 追加

条件式ビルダー

正常イベントの生成 条件式 復旧条件式 なし

*復旧条件式 {HostA:log[/hoge/moga.log,ERROR].nodata(30)}=1 追加

条件式ビルダー

障害イベント生成モード 単一 複数

正常時のイベントクローズ すべての障害 タグの値が一致したすべての障害

手動クローズを許可

復旧イベントを
正しく生成することが大事！

ちょっと宣伝

手動での障害復旧って大変ですよね…

- ✓ 特にログやTrapのバースト時は確認や復旧に時間がかかる
- ✓ そもそも通知が多くて重要な障害を見落とす可能性もある



アラートの集約機能&自動クローズ可能なツール
「**GatherAlert**」があります！

何ができるの？

一定期間おきに、障害内容をまとめて通知できる

| ! | 📄 | 📧 | 差出人 | 件名 | 受信日時 |
|---|---|---|---------|---------|----------------------|
| ▼ | | | | | 今日 |
| 📧 | | | Gath... | Gath... | 2021/01/08 (金) 16:03 |
| 📧 | | | Gath... | Gath... | 2021/01/08 (金) 15:53 |
| 📧 | | | Gath... | Gath... | 2021/01/08 (金) 15:43 |
| 📧 | | | Gath... | Gath... | 2021/01/08 (金) 15:33 |
| 📧 | | | Gath... | Gath... | 2021/01/08 (金) 15:23 |
| 📧 | | | Gath... | Gath... | 2021/01/08 (金) 15:13 |
| 📧 | | | Gath... | Gath... | 2021/01/08 (金) 15:03 |
| 📧 | | | Gath... | Gath... | 2021/01/08 (金) 14:53 |

GatherAlert【重度・致命的障害】



GatherAlert@zabicom.com

宛先 ●



attach.zip

712 バイト

監視担当者様

ZABICOM GatherAlert です。

障害を検知しました。

お客様名：(株)南畑エンターテイメント様

システム名：猿楽マネジメントシステム

以下、障害内容：

```
"イベント発生日時","イベント深刻度","イベント種別","ホスト名","イベント内容","[他件数]"  
"2021/01/08 15:50:23","重度の障害","正常","appservice02","Ping 監視【ga_app3】"  
"2021/01/08 15:51:38","致命的な障害","正常","httpservice01","Ping 監視【gahttp3】"  
"2021/01/08 15:54:14","重度の障害","障害","httpservice01","Web 監視【gahttp1】"  
"2021/01/08 15:56:14","重度の障害","正常","httpservice01","Web 監視【gahttp1】"  
"2021/01/08 15:58:53","重度の障害","障害","appservice02","Ping 監視【ga_app3】"
```

以上

何ができるの？

GatherAlert【重度・致命的障害】

GatherAlert@zabicom.com
宛先

attach.zip
712 バイト

csv出力可能！

監視担当者様
ZABICOM GatherAlert です。
障害を検知しました。
お客様名：(株)南畑エンターテイメント様
システム名：猿楽マネジメントシステム
以下、障害内容：

"イベント発生日時","イベント深刻度","イベント種別","ホスト名","イベント内容","[他件数]"
"2021/01/08 15:50:23","重度の障害","正常","appservice02","Ping 監視【ga_app3】"
"2021/01/08 15:51:38","致命的な障害","正常","httpservice01","Ping 監視【gahttp3】"



通知イメージ (Teams)

The screenshot shows the Microsoft Teams interface. At the top, there is a search bar with the text "検索". Below it, the left sidebar contains navigation icons for "アクティビティ", "チャット", "チーム", "会議", "通話", "ファイル", and "アプリ". The main area displays a channel named "障害通知 (平日)". The notification message is titled "【障害通知】" and contains the following text:

以下、障害内容

"イベント発生日時","イベント深刻度","イベント種別","ホスト名","イベント内容","[他件数]"
"2021/03/17 12:20:08","致命的な障害","障害","appservice01","CPU使用率【gasvc3】"
"2021/03/17 12:20:09","重度の障害","障害","appservice01","プロセス監視【gasvc2】"
"2021/03/17 12:23:59","軽度の障害","障害","appservice02","プロセス監視：プロセス【ga_app1】"
"2021/03/17 12:24:09","軽度の障害","障害","httpservice01","SNMPPolling監視：【gahttp2】からlinkアラートを検知"
"2021/03/17 12:25:14","警告","障害","appservice01","プロセス監視：httpプロセスダウンアラート0"
"2021/03/17 12:26:14","重度の障害","障害","httpservice01","Web監視【gahttp1】"
"2021/03/17 12:27:24","重度の障害","障害","appservice02","CPU空き率【ga_app2】"
"2021/03/17 12:28:53","致命的な障害","障害","appservice02","Ping監視【ga_app3】"
"2021/03/17 12:28:59","軽度の障害","障害","appservice02","プロセス監視：プロセス【ga_app1】"

以上

At the bottom of the notification, there is a "返信" (Reply) button and a "新しい投稿" (New Post) button. The bottom status bar shows "outlook.office.com を待機しています..."

通知イメージ (Slack)

The screenshot shows a Slack channel interface. On the left is a sidebar with the channel name 'GatherAlert' and a list of channels including '# 4t-a'. The main area shows a chat window for '# 4t-a' with two messages from the 'GatherAlert_Notification' app. Each message is a system alert with a title in bold: '【障害通知】 【重度・致命的障害】'. The messages contain Japanese text identifying the customer and system, followed by a list of event details in a structured format. The interface includes a top navigation bar with a date 'Wednesday, July 21st' and a bottom message input area with various formatting and action icons.

GatherAlert # 4t-a

+ Add a bookmark

以上

Wednesday, July 21st

GatherAlert_Notification APP 1:40 PM

【障害通知】 【重度・致命的障害】

監視担当様

ZABICOM GatherAlertです。障害を検知しましたのでお知らせ致します。

お客様名：(株)南畑エンターテイメント様

システム名：猿楽マネジメントシステム

以下、障害内容：

"イベント発生日時";"イベント深刻度";"イベント種別";"ホスト名";"イベント内容";"[他件数]"

"2021/07/21 00:40:02";"重度の障害";"正常";"appservice01";"ログ監視";"他4件"

"00:40:02";"重度の障害";"障害";"appservice01";"ログ監視";"他4件"

"2021/07/21 00:40:40";"致命的な障害";"正常";"appservice02";"Ping監視【ga_app3】"

"00:40:54";"重度の障害";"障害";"appservice01";"プロセス監視【gasvc2】"

"00:42:39";"重度の障害";"障害";"appservice02";"CPU空き率【ga_app2】"

"2021/07/21 00:45:39";"重度の障害";"正常";"appservice02";"CPU空き率【ga_app2】"

"00:49:10";"致命的な障害";"障害";"appservice02";"Ping監視【ga_app3】"

以上

GatherAlert_Notification APP 1:50 PM

【障害通知】 【重度・致命的障害】

監視担当者様

ZABICOM GatherAlertです。障害を検知しましたのでお知らせ致します。

お客様名：(株)南畑エンターテイメント様

システム名：猿楽マネジメントシステム

以下、障害内容：

"イベント発生日時";"イベント深刻度";"イベント種別";"ホスト名";"イベント内容";"[他件数]"

Send a message to #4t-a

⚡ B I ↻ ↵ 🔗 📄 📄 📄 📄 Aa @ 😊 📎 ▶

通知イメージ (LINE)



すべて 友だち グループ 公式アカウント オープンチャット

トークルームとメッセージ検索

LINE Notify 午後 12:23
[GatherAlert通知]【障害通知】 以下、障害内容 "イ...

2021/03/17 12:15:08, 致命的な障害, "障害", "httpservice01", "Ping監視【gahttp3】"
"2021/03/17 12:18:53", "致命的な障害", "障害", "appservice02", "Ping監視【ga_app3】"
"2021/03/17 12:18:59", "軽度の障害", "障害", "appservice02", "プロセス監視：プロセス【ga_app1】"
"2021/03/17 12:19:14", "重度の障害", "障害", "httpservice01", "Web監視【gahttp1】"
以上 午後 12:13

[GatherAlert通知]【障害通知】
以下、障害内容
"イベント発生日時", "イベント深刻度", "イベント種別", "ホスト名", "イベント内容", "[他件数]"
"2021/03/17 12:20:08", "致命的な障害", "障害", "appservice01", "CPU使用率【gasvc3】"
"2021/03/17 12:20:09", "重度の障害", "障害", "appservice01", "プロセス監視【gasvc2】"
"2021/03/17 12:23:59", "軽度の障害", "障害", "appservice02", "プロセス監視：プロセス【ga_app1】"
"2021/03/17 12:24:09", "軽度の障害", "障害", "httpservice01", "SNMPPolling監視：【gahttp2】からlinkアラートを検知"
"2021/03/17 12:25:14", "警告", "障害", "appservice01", "プロセス監視：httpプロセスダウンアラート0"
"2021/03/17 12:26:14", "重度の障害", "障害", "httpservice01", "Web監視【gahttp1】"
"2021/03/17 12:27:24", "重度の障害", "障害", "appservice02", "CPU空き率【ga_app2】"
"2021/03/17 12:28:53", "致命的な障害", "障害", "appservice02", "Ping監視【ga_app3】"
"2021/03/17 12:28:59", "軽度の障害", "障害", "appservice02", "プロセス監視：プロセス【ga_app1】"
以上 午後 12:23

メッセージを入力

📎 📌 🗑️

8:35 15:50

< 10 + LINE Notify 🔍 📄 ☰

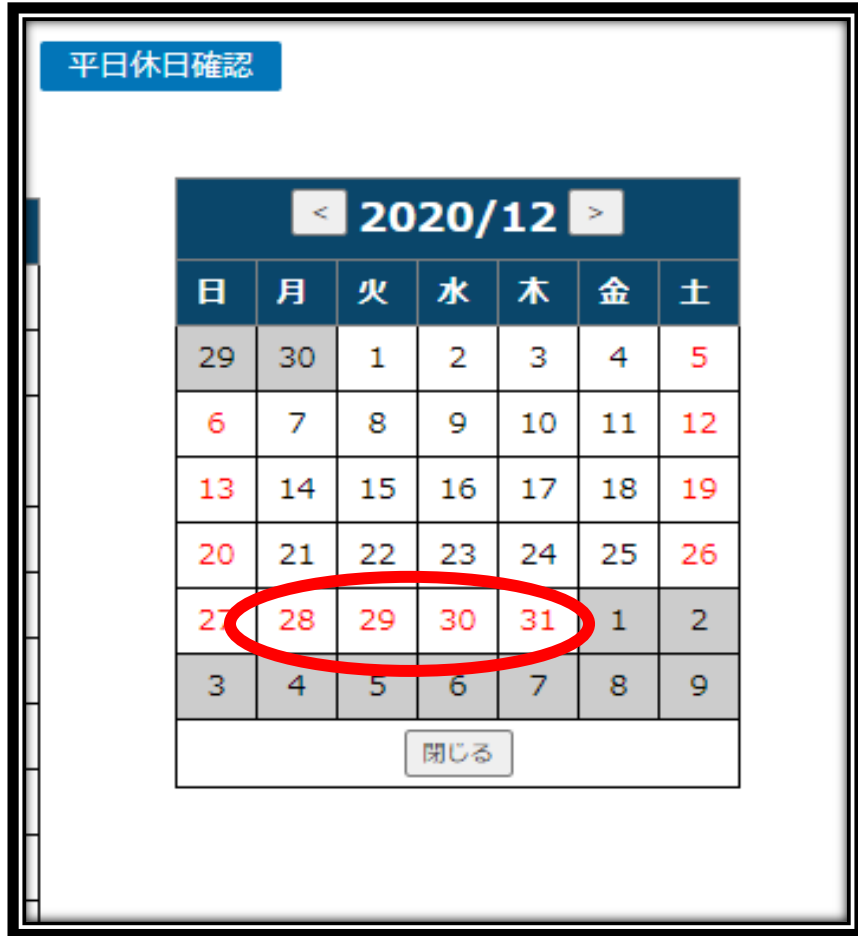
[GatherAlert通知]【障害通知】【警告・軽度の障害】
15:50

監視担当様
ZABICOM GatherAlertです。障害を検知しましたのでお知らせ致します。
お客様名：株南畑エンターテイメント様
システム名：猿楽マネジメントシステム
以下、障害内容：
"イベント発生日時", "イベント深刻度", "イベント種別", "ホスト名", "イベント内容", "[他件数]"
"02:50:01", "警告", "障害", "appservice01", "ログ監視：【/var/log/messages】メッセージを検知【Jul 21 15:50:01 appservice01 systemd: Started Session 60672 of user root.】", "他17件"
"2021/07/21 02:50:38", "軽度の障害", "正常", "appservice02", "プロセス監視：プロセス【ga_app1】"
"02:54:08", "軽度の障害", "障害", "appservice02", "プロセス監視：プロセス【ga_app1】"
"2021/07/21 02:55:38", "軽度の障害", "正常", "appservice02", "プロセス監視：プロセス【ga_app1】"
"02:55:53", "警告", "障害", "appservice01", "プロセス監視：httpプロセスダウンアラート0"
"02:59:08", "軽度の障害", "障害", "appservice02", "プロセス監視：プロセス【ga_app1】"
以上 15:50

+ 📷 🖼️ Aa 😊 🗣️

さらに

営業日と営業時間を自由に設定することができる



カレンダー機能
実装してますが何か？



どんなメリットがある？

1. アラートがまとまって通知される
2. 営業日・営業時間に応じて通知先を振り分けられる
3. 通知した障害については**自動でクローズ**する



| 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 | 確認済 | アクション |
|----------|---------------------------------|----------|-------|----|---------------|--------------------------------------|--------|-----|-------|
| 07:54:14 | <input type="checkbox"/> 重度の障害 | 07:56:14 | 解決済 | | httpservice01 | Web監視【gahttp1】 | 2m | はい | 1 3 |
| 07:53:59 | <input type="checkbox"/> 軽度の障害 | 07:55:29 | 解決済 | | appservice02 | プロセス監視：プロセス【ga_app1】 | 1m 30s | はい | 1 3 |
| 07:50:39 | <input type="checkbox"/> 軽度の障害 | 07:52:39 | 解決済 | | httpservice01 | SNMPPolling監視：【gahttp2】からlinkアラートを検知 | 2m | はい | 1 3 |
| 07:50:38 | <input type="checkbox"/> 致命的な障害 | 07:59:08 | 解決済 | | appservice01 | CPU使用率【gasvc3】 | 8m 30s | はい | 1 3 |
| 07:49:38 | <input type="checkbox"/> 致命的な障害 | 07:51:38 | 解決済 | | httpservice01 | Ping監視【gahttp3】 | 2m | はい | 1 3 |
| 07:48:59 | <input type="checkbox"/> 軽度の障害 | 07:50:29 | 解決済 | | appservice02 | プロセス監視：プロセス【ga_app1】 | 1m 30s | はい | 1 3 |
| 07:48:53 | <input type="checkbox"/> 致命的な障害 | 07:50:23 | 解決済 | | appservice02 | Ping監視【ga_app3】 | 1m 30s | はい | 1 3 |
| 07:47:14 | <input type="checkbox"/> 重度の障害 | 07:49:14 | 解決済 | | httpservice01 | Web監視【gahttp1】 | 2m | はい | 1 3 |
| 07:43:59 | <input type="checkbox"/> 軽度の障害 | 07:45:29 | 解決済 | | appservice02 | プロセス監視：プロセス【ga_app1】 | 1m 30s | はい | 1 3 |
| 07:42:24 | <input type="checkbox"/> 重度の障害 | 07:45:24 | 解決済 | | appservice02 | CPU空き率【ga_app2】 | 3m | はい | 1 3 |

まとめ

- ✓ 何でAll in Oneコンテナで実装したの？
- ✓ problemテーブル肥大化に関する事例と対応について
- ✓ GatherAlertの紹介



ご聴講ありがとうございました！