



# サーバー監視のはなし

監視と通知と障害と俺とお前と大五郎

水野源

Ubuntu Japanese Team



監視

とは



# サーバー監視

サーバー障害は**絶対に起こるもの**  
かといって24時間有人監視もなかなかできないよね  
そこで

- サービスダウンを検知して通知したり
- リソースの変化を記録して分析に役立てたり
- リソースの変化から将来起こりそうな障害を予測したり

そんなツールの話をします



# 監視ツールいろいろ

これひとつで完璧、という定番はなさげ

- Zabbix
- Nagios
- Sensu
- Munin
- ほかにたくさん



# 外形監視と内部監視

監視は大きく分けて外と内のふたつ

## 外形監視

外部から見た接続状況を監視する

## 内部監視

システムの内部状態を監視する



# サービスとリソース監視

内部監視はサービスとリソースに分けて考えられる

## サービス監視

サービスの稼働状況を監視する  
ポートは空いてる？  
プロセスは起動してる？

## リソース監視

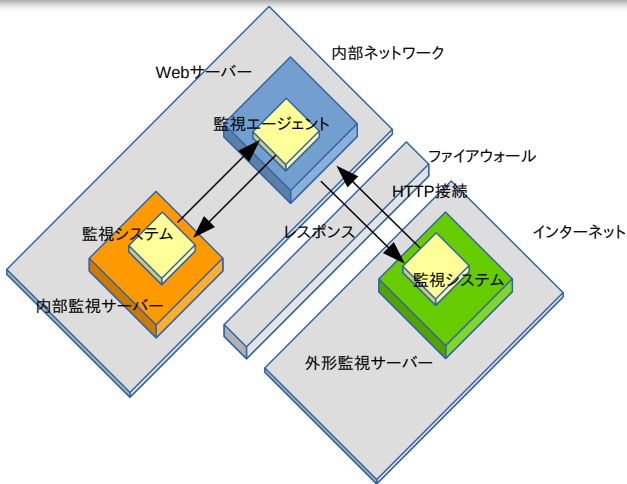
リソースの変化を記録する  
CPU負荷は？  
空きメモリは？



# どこから監視すればいい？

- 外形監視はユーザーと同じ場所(外)にいる必要がある
- リソース監視は内側からのアクセスが必要
  - 単一の監視サーバーで両立は無理
- → **外部と内部にそれぞれ監視サーバーが必要**

# 概念図







# わかった。では何を使う？

監視する対象に合ったシステムを選択しよう

## リソース監視がしやすいシステム

Munin  
Mackerel とか

## 死活監視がしやすいシステム

Nagios  
Monit とか

# リソース の監視



# リソース監視ツール

継続的なメトリックの収集と可視化を行うツール

サーバーのリソースの変化を監視記録することで、**障害を事前に回避する**、あるいは障害発生後の**分析に役立てる**

# たとえばMunin



Overview :: Ubuntu :: [trusty.mizuno-as.net](#)  
trusty.mizuno-as.net :: [ disk network processes system time ]

## Problems

Critical (0)  
Warning (0)  
Unknown (0)

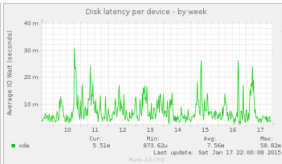
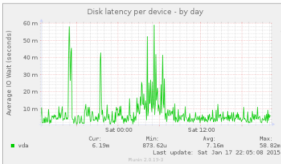
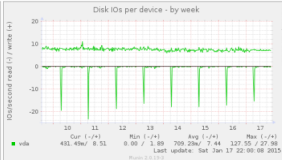
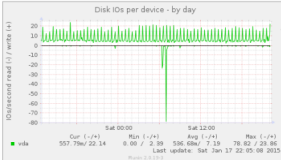
## Groups

Debian  
Ubuntu

## Categories

disk [ d w m y ]  
network [ d w m y ]  
postfix [ d w m y ]  
processes [ d w m y ]  
system [ d w m y ]  
time [ d w m y ]

## disk



Muninのグラフ



# Muninとは

- リソース監視に特化したシステム
  - CPU負荷、メモリ使用量、トラフィックなどをグラフ化

詳しくは

[Ubuntu Weekly Recipe 第359回 Muninでサーバーのリソースを可視化しよう](#)参照



# Muninに足りないもの

とはいえ

- あんまりモダンじゃない
- 値が記録されるけど、言ってしまうえばそれだけ
- アラートも出せるけど、ぶっちゃけ使いづらい
- ノードの追加にひと手間必要

# そこでMackerel

The screenshot shows the Mackerel monitoring interface for a host named 'trusty'. The interface is organized into several sections:

- Navigation:** A sidebar on the left contains 'Dashboards', 'Overview', 'Hosts', 'Services', 'VPS', 'Monitors', and 'Alerts'. The 'Hosts' section is currently selected, showing 'mizuno'.
- Host Information:** The main header displays the host name 'trusty' with a '設定' (Settings) icon. Below it, a 'working' status indicator is shown.
- System Metrics:** A central section titled 'システムメトリック' (System Metrics) features a 'loadavg5' line graph and a 'cpu' table. The CPU table shows: idle (176.9% / 200.0%), guest (0.0% / 200.0%), steal (0.0% / 200.0%), lowait (14.8% / 200.0%), and user (5.3% / 200.0%).
- Hardware Information:** A 'ハードウェア' (Hardware) section lists: 993MB RAM, 2 x Westmere E56xx/L56xx/X56xx (Nehalem-C), and GNU/Linux 3.13.0-48-generic.
- Monitors:** A '4 Monitors' section on the right lists: connectivity, CPU % (with a red warning bar at 79% / 95%), Memory % (with a red warning bar at 80% / 95%), and Filesystem % (with a red warning bar at 79% / 80%).



# Mackerelとは

- はてなの有償サービス
- 見やすいグラフ
- アラートも出せる
- 外形監視はオマケかな
- けっこう安い
- サーバー5台までなら無料プランでもいける





# システムメトリック

## ■ デフォルトで収集されるメトリック

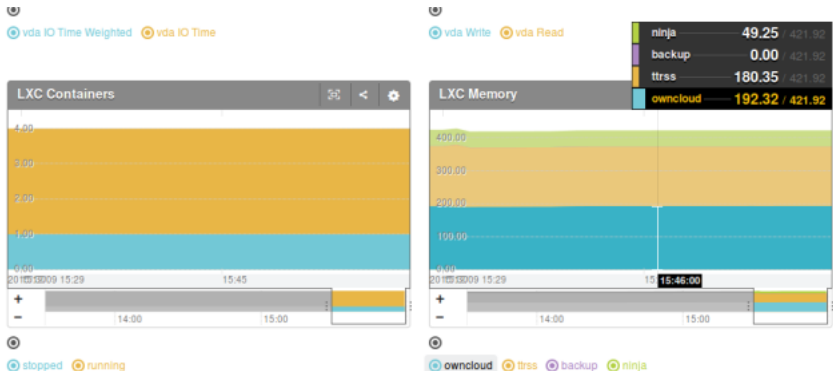
- ロードアベレージ
- CPU利用率
- メモリ使用量
- IOPS
- ディスク使用量
- ネットワークトラフィック



# カスタムメトリック

- サーバーごとに自由に追加できるメトリック
- プラグインで実装
  - まずは公式プラグイン集を入れよう
- 決まった出力ができれば内部実装は好き勝手にOK
- フォーマットはSensu互換
- 簡単に作れるよ

# カスタムメトリックの例



LXCコンテナの数と消費メモリを可視化



# サービスマトリック

- 個別のサーバーに関連づかないメトリック
- たとえばサービス全体の売り上げを可視化



# URL外形監視

- http/httpsのみ
- 無料プランでは使えない

Host Metric   Service Metric   External Http

 **External Http Monitor**  
指定したURLに対して1分毎にステータスコードの監視を行います。

外形監視を利用するにはプランをアップグレードする必要があります。

[より詳しく](#)















# Mackerelでの監視と通知

おおまかな手順は以下の通り

1. 監視するメトリックを設定
2. 監視項目ごとにしきい値を設定
3. アラートの送信先を設定

# 監視ルールを追加

## 監視ルール (4)

 connectivity	connectivity	<input checked="" type="checkbox"/> すべて	 
 CPU %	<span>&gt; 70%</span> <span>&gt; 90%</span> CPU %	<input checked="" type="checkbox"/> すべて	 
 Filesystem %	<span>&gt; 70%</span> <span>&gt; 80%</span> Filesystem %	<input checked="" type="checkbox"/> すべて	 
 Memory %	<span>&gt; 80%</span> <span>&gt; 99%</span> Memory %	<input checked="" type="checkbox"/> すべて	 



# 障害発生

事前に設定した通知チャンネル経由でアラートが送信される

## Alerts ⚙️ 設定

### In Progress (1) ▶



In Progress Critical

2015/12/09 16:17:26

CPU %

100.0 % > 90%

working



trusty

VPS: web







# 通知チャンネル

- メール
- Slack
- HipChat
- PagerDuty
- ChatWork

などなど

Slackに直接投げられたりするところが今風

# メール通知の例



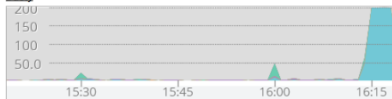
Alert



## mizuno: trusty

In Progress

trusty



Now In Progress — CPU %

16:18 Critical 100.0% > 90.0%

16:17 Warning 77.15% > 70.0%

12/9

16:17 Opened — CPU %



# 障害復旧

## In Progress ▶



現在、アラートはありません

監視ルールでアラートの発生条件を設定できます

## Closed ▶



Closed

CPU %

working



trusty

VPS: web

2015/12/09 16:17:26



# Mackerelまとめ

- グラフは見やすい
  - 有料で1年、無料だと1日だけ
- 色々な意味で今風
- 国産
- 総じて使いやすいよ
- 死活監視はできるけどちょっと弱いかな

死活  
監視



# 死活監視ツール

- サービスが活着ているかを監視
- 死んだらアラート
  - 外部から監視するところがミソ
  - Pingdomとか有名
- リソース変化を記録できるものもあるけど、主ではない



# Xymon

## BigBrotherの系譜に連なる由緒正しい(?)監視システム

The screenshot shows the Xymon web interface. At the top, there are navigation tabs: Views, Reports, Administration, and Help. The main header displays 'Xymon', 'Current Status', and the date/time 'Thu Dec 10 02:34:17 2015'. Below the header is a horizontal menu with links for various system metrics: bbd, clientlog, conn, cpu, disk, files, http, info, inode, memory, msgs, ports, procs, trends, xymond, xymongen, and xymonnet. The main content area shows the host 'xymon.mizuno-as.net' followed by a row of 17 status indicators. Each indicator consists of a small green circle with a white icon inside, representing the status of a specific metric. The icons include a solid circle, a diamond, a smiley face, a sad face, a warning sign, and a diamond with a dot. In the bottom right corner, the version number 'Xymon 4.3.21' is displayed.



# Xymonの特徴

- 外形監視と内部監視の双方に対応
- テキストで設定できる
- アラート時にスクリプトを叩けるのでなんでもできちゃう
- 監視を一時的に止める、がやりやすい
- **外形監視とヒストリーがお手軽な上に強力**

詳しくは[Ubuntu Weekly Recipe 第383回 Xymonではじめるサーバーモニタリング](#)参照





# Xymonで外形監視

- サービス名を指定するだけで死活を監視できる
- http/https/ssh/smtp/pop/imap/ftp/ldapなどなど
  - httpはステータスコードでの監視もできる
- httpsは**SSL証明書の有効期限も自動的に監視**



# Xymonで内部監視

- プロセスが起動してるか
- ポートは開いているか
- ログにエラーは出ていないか
- リソース監視全般
  - Muninのようなリソース監視としても利用できる
  - クライアントにスクリプトを追加することで任意に拡張可能



# その時何があったか

## 障害あるある

後からグラフを見ると負荷が上がったことが判明したが、**どんなプロセスが動いていたのか調べようがない**



# XymonのHistory機能

Xymonはステータスが変化した時のシステム情報をダンプして記録してくれる

```
Thu Dec 10 03:24:28 UTC 2015 up: 01:10, 1 users, 99 procs, load=6.68
```

```
! Load is HIGH
System clock is 0 seconds off
```

```
top - 03:24:35 up 1:10, 1 user, load average: 8.01, 6.68, 3.84
Tasks: 97 total, 10 running, 87 sleeping, 0 stopped, 0 zombie
%Cpu(s): 24.7 us, 0.5 sy, 0.0 ni, 63.7 id, 0.2 wa, 0.0 hi, 0.0 si, 10.9 st
KiB Mem: 601568 total, 330936 used, 270632 free, 20320 buffers
KiB Swap: 0 total, 0 used, 0 free. 219772 cached Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4678	ubuntu	20	0	7344	96	0	R	51.0	0.0	0:52.91	stress
4675	ubuntu	20	0	7344	96	0	R	46.7	0.0	0:52.76	stress
1	root	20	0	37448	5464	3952	S	0.0	0.9	0:04.65	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/0



# Xymonまとめ

- 主要なサービスを手軽に外形監視できるのが嬉しい
- 証明書監視も地味に便利
- メトリックだけだと、障害が起きたことしかわからない
- **その瞬間のpsやnetstatが記録される**のが超便利
- 内と外のXymon2台体制もおすすめ



# ざっくりまとめ

## Mackerel

メトリックの見やすさはとてもよい  
Muninプラグインをそのままラップできる点も魅力  
無料プランで試してみるのすすめ

## Xymon

世間ではマイナーだけどかなり使える  
ドワンゴさんも使ってるらしい  
お手軽に自前で外形監視したいなら超おすすめ

障害発生



# アラート通知について

## 障害あるある

監視システムが障害を検出しても、人間が気づかない

では**いかにして通知するか？**





# メールで通知

## ■ やっぱり基本

- 普段は揮発性の通知(push/チャット)の方が便利かな
- どちらかというと記録を残す意味が大きいかも

## ■ メール着信程度じゃ**夜中起きねえよ**

- → 確実に起きるための工夫を試してみる



# チャットで通知

- スピード大事
- 普段の業務中はもはやメールなんか見ない
- Slack連携も今となっては基本



# スクリプト連携

- アラートをトリガーに任意のスクリプトを実行
  - 弊社ではゆっくり連携
- Xymonはこのへんが柔軟に設定できてとても便利

# 通知管理 サービス



# PagerDuty

- 最強の連絡手段は電話
  - SMSでも通知を受けたい
  - スマホnoPush通知でも受けたい
- 当番を決めたい
- エスカレーションしたい

といった、**通知のしかたを管理するシステム**

# アラートを集約して通知

監視システムからアラートを受け取り、事前に設定した相手に、任意の手段、スケジュールで通知を送れる




Hajime Mizuno

Contact Information

Notification Rules

User Settings

## When a high-urgency incident is assigned to me...

 **Immediately** after it's assigned to me, phone me at +81 [REDACTED] (Mobile)



**Immediately** after it's assigned to me, push notify me on ASUS\_Z00AD



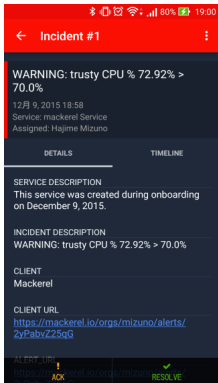
 **Immediately** after it's assigned to me, SMS me at +81 [REDACTED] (Mobile)



[+ Add Notification Rule](#)



# スマホからack/resolv



# 弊社の 事例





# Skype Bot連携

- 弊社製のSkype Bot
- 通称「ゆっくり」
- もともとはLinux版Skypeクライアント+PHP
- 今はHubotのSkypeWebAdapterで実装

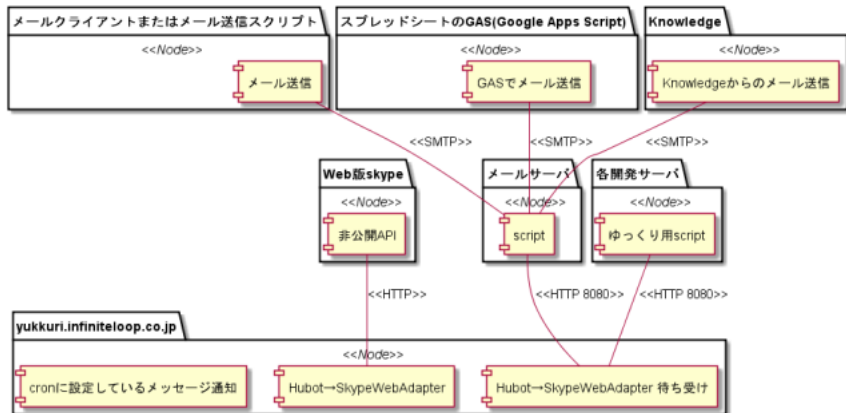


# ゆっくりの機能

- 任意のテキストを発言する
- 任意の音声を発生する
- スケジュールリマインダ
- 様々なコマンドで情報発信

# ゆっくりアーキテクチャ

ゆっくり配置図





# 障害が発生したら

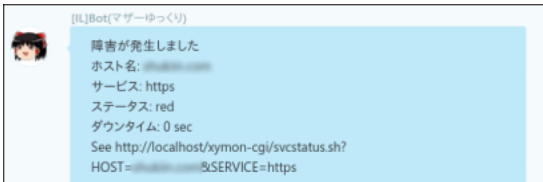
アラートをトリガーにXymonがスクリプトをキック

```
#!/bin/sh

URL=$(echo $BBALPHAMSG | /bin/grep -o -e 'See .*')

if [ $RECOVERED -ne 1 ]; then
  /usr/local/bin/sendMessage $RCPT "障害が発生しました¥n
  ホスト名: $BBHOSTNAME¥n
  サービス: $BBSVCNAME¥n
  ステータス: $BBCOLORLEVEL¥n
  ダウンタイム: $DOWNSECS sec¥n
  $URL"
fi
```

# チャットに発言



- それと同時に社内へ音声アナウンス
- ちなみにパトライトも回す



# まとめ

監視運用は

- 異常を**検出**できること
- 異常を**記録**できること
- 異常に**気づ**けること

どれが欠けてもダメ

それぞれに適したサービスを!