



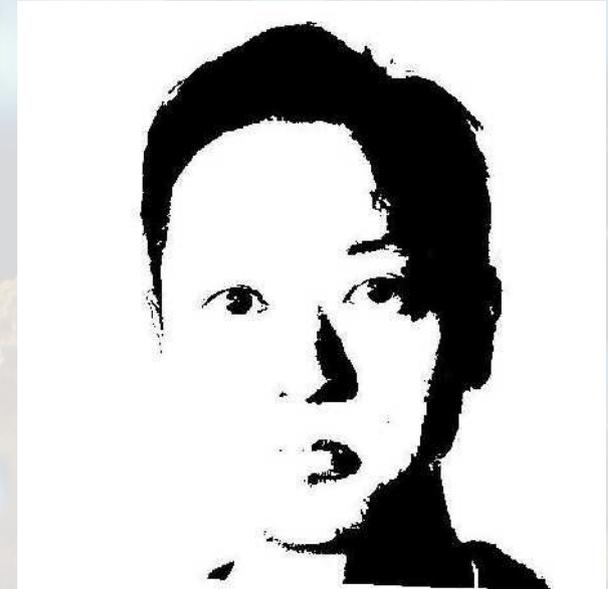
ZabbixにおけるDocker監視の方法

株式会社 アーベルソフト
渡邊 隼人

自己紹介

Aboutme:

Name: Watanabe Hayato ,
Fammiry: [妻:1, 娘:1, 息子:1] ,
Hobby: アクアリウム、ものづくり



Account:

Twitter: @_BSmile_ ,
Facebook: 渡邊 隼人 ,
aboutme: <http://about.me/bsmile/>

Company:

Name: 株式会社 アーベルソフト※注 ,
Address: 埼玉県坂戸市芦山町 22-13 ,
Number of mployees: 47 (2015/08)

おことわり

ここで発表する内容は現在所属する会社の**公式見解を示すものではありません**

また、今回使用したDockerはServer/Clientともに1.8.3となっています。

予めご了承ください

Agenda

- **Docker**について
- 一般的に考えるZabbixでの**コンテナ監視**
- Zabbixでの**収集ツール作ってみました**
- 本格的に作りこもうとしたある日...
- 作ってみてわかったこと(**まとめ**)
- OSS活動サークルの宣伝(非営利)

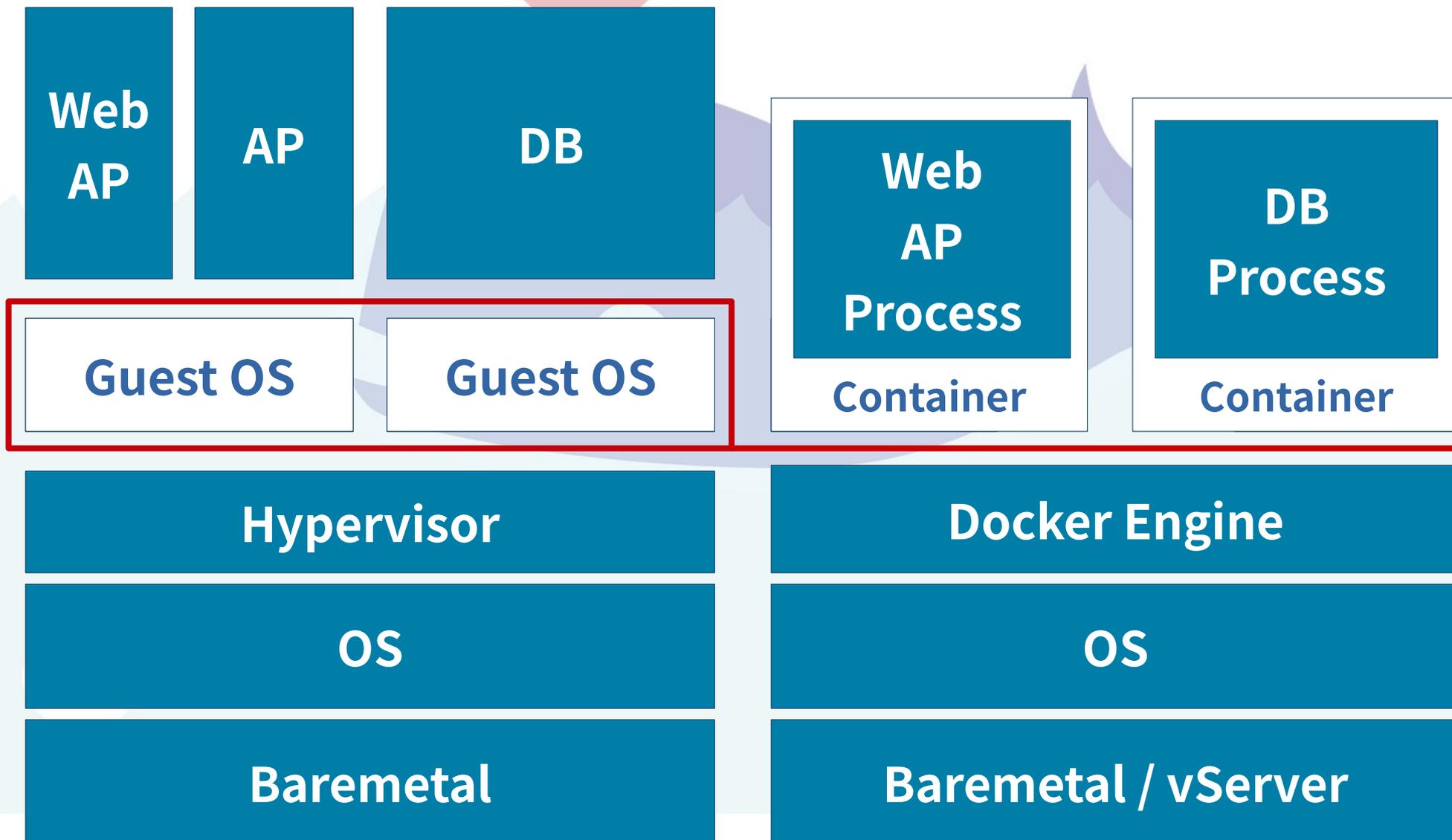
A large whale is breaching the water surface, viewed from below. The whale's head and back are visible above the water, while its tail is still submerged. The water is dark blue, and the whale's skin is wet and glistening. The whale's tail is a light color, contrasting with the dark water.

Dockerについて

[1/5]

Dockerについて

仮想サーバとの違い(LXC / namespace)



Dockerについて

Z

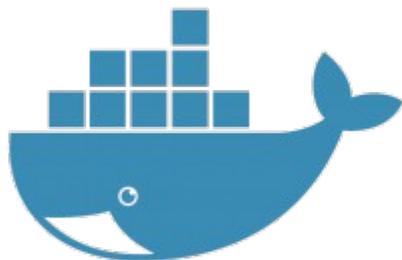
Container

Cgroup

Root fs
Storage Driver

Docker Engine

Docker Host



Cgroup(コントロールグループ)

プロセスグループのCPU、メモリ、ネットワークのリソース使用状態の確認・制御が可能なカーネルの機能

Zabbixに関わらず、コンテナリソース監視はCgroupをチェックする

Dockerについて

Z

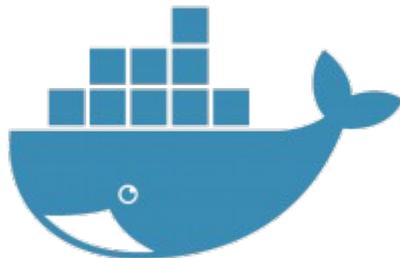
Container

Cgroup

Root fs
Storage Driver

Docker Engine

Docker Host



Storage Driver

• Dockerでは様々なStorageDriverを選択する事が可能。

- devicemapper

- aufs

- btrfs

- zfs

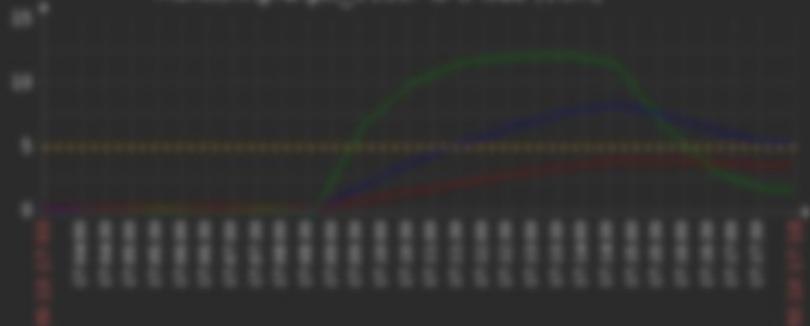
...

※ RHEL7系のデフォルトDriverは devicemapper

一般的に考えるZabbixでのコンテナ監視

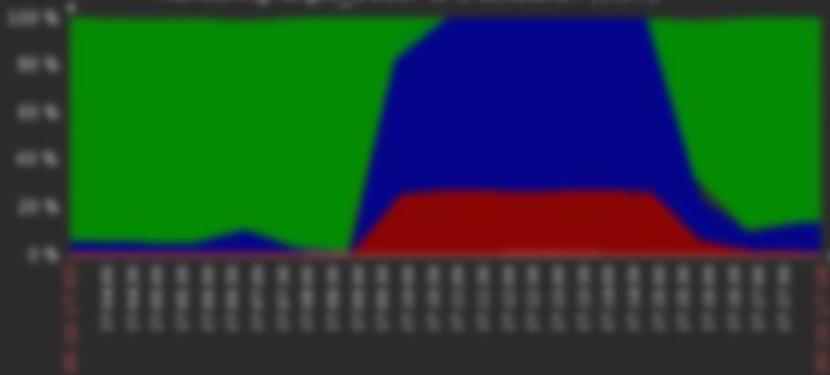
[2/5]

MonitoringTarget_0100: CPU load (15m)



	avg	min	max	total
Processor load (1 min average per core)	1.74	0.24	7.77	1212
Processor load (5 min average per core)	1.24	0.24	5.19	813
Processor load (15 min average per core)	1.14	0.23	3.82	588
Trigger: Processor load is high on MonitoringTarget_0100 (2x 5)				

MonitoringTarget_0100: CPU utilization (15m)



	avg	min	max	total
CPU idle time	88.14%	71%	95.17%	98.81%
CPU user time	11.14%	0.81%	15.17%	15.17%
CPU system time	0.71%	0.1%	1.17%	1.17%
CPU wait time	0.17%	0.01%	0.17%	0.17%
CPU nice time	0%	0%	0%	0%
CPU steal time	0%	0%	0%	0%

MonitoringTarget_0100: Memory usage (15m)



	avg	min	max	total
Available memory	467.74 MB	425.91 MB	468.94 MB	467.74 MB

MonitoringTarget_0100: Swap usage (15m)



	avg	min	max	total
Total swap space (in MB)				
Free swap space (in MB)				

MonitoringTarget_0100: Number of processes (15m)



	avg	min	max	total
Number of processes	25	10	100	100

MonitoringTarget_0100: Number of running processes (15m)



	avg	min	max	total
Number of running processes	7	7	43	11



こう考える方が多いと思います。





コンテナって言っても普通のOSでしょ？
ならZabbixエージェントを
インストールすれば良いんじゃない？

DockerHostとコンテナ→ZabbixAgent

1) 物理サーバへDockerとZabbixServer/ZabbixAgentインストール

2) コンテナへZabbixAgentインストール

3) スクリーンを監視し、CPU負荷によるグラフの推移を確認

負荷①：  コンテナで実施

負荷②：  Dockerホストで実施

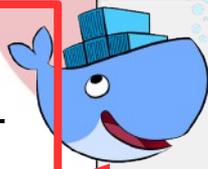
opensslでのCPU負荷テスト

```
# openssl speed -multi `grep processor /proc/cpuinfo|wc -l`
```

実際にAgent監視を実施してみると

コンテナへの負荷

→ Dockerホスト・コンテナ
同じデータ値が取れる

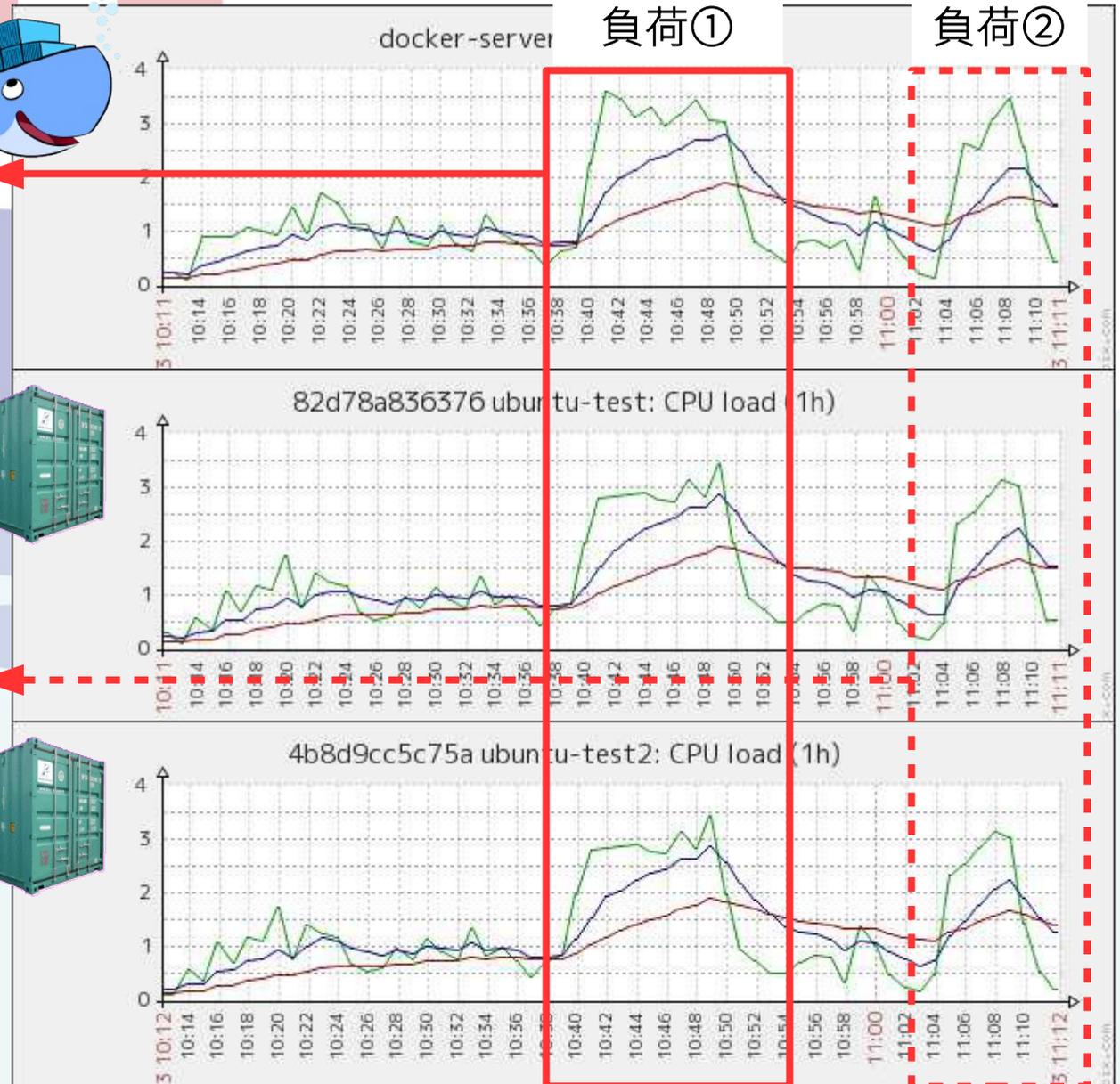


Dockerホストへの負荷

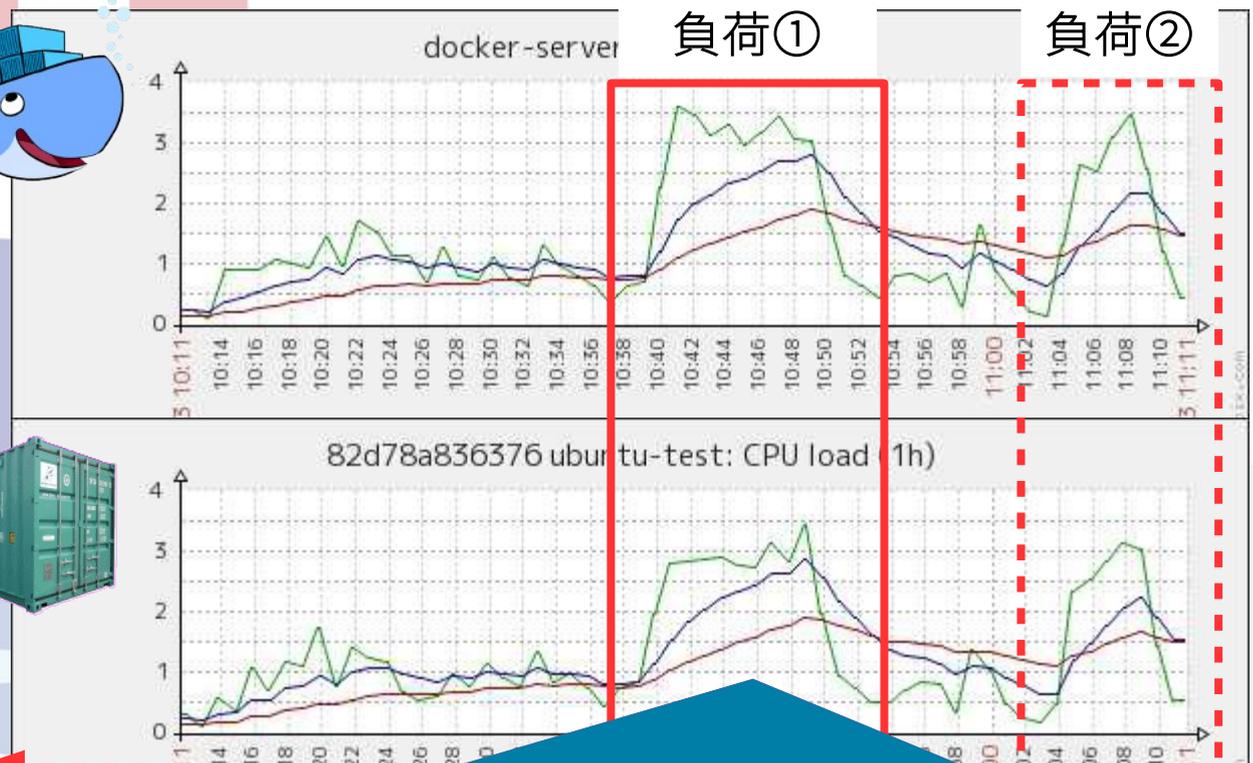
→ Dockerホスト・コンテナ
同じデータ値が取れる



注：グラフにゆらぎがあるのは、取得間隔1分としている為、秒間で取得値に若干の誤差がある為



実際にAgent監視を実施してみると



コンテナ上で監視するCPU・メモリ情報はDockerホスト上と同じLinuxシステムリソース情報を取得している為、ZabbixAgentを互いにインストールした監視は重複しあまり意味が無い？

コンテナへAgentを入れDockerホストのリソース監視をすることも。



DISK監視が非常に面倒



Storage Driverごとデータ使用量確認

DeviceMapperは1つのディスクイメージファイルをデバイスプールとし、複数コンテナで共有している

```
[root@develop1 ~]# ll /dev/mapper/  
lrwxrwxrwx 1 root root    7 11月  9 16:08 docker-253:3-919183-  
4b8d9cc5c75ab3573b87461cca7a529959621975966592d7466febe451ad4950 -> ../dm-2  
lrwxrwxrwx 1 root root    7 11月 14 09:06 docker-253:3-919183-  
82d78a836376c05705908ef0e5badbb510bdb41e81a45e1cf59870caa96d9054 -> ../dm-1  
lrwxrwxrwx 1 root root    7 11月 15 05:14 docker-253:3-919183-  
8a894448c8d5a593b46c5bb8d50fb9989ea6bcf088c6dd1c51b3004495e6bc5e -> ../dm-3  
lrwxrwxrwx 1 root root    7 11月  9 15:17 docker-253:3-919183-pool -> ../dm-0
```



Storage Driverごとデータ使用量確認

devicemapperのコンテナ使用量確認には
dmをマウントするかdocker execで内部から容量を
確認する必要がある(お薦めしない)

```
[root@develop1 ~]# docker exec ubuntu-test2 df --output="used" /
```

```
Used
```

```
286396
```

```
[root@develop1 ~]# mount /dev/mapper/docker-253\:3-919183-
```

```
4b8d9cc5c75ab3573b87461cca7a529959621975966592d7466febe451ad4950 /mnt/
```

```
[root@develop1 ~]# df --output="used" /mnt/
```

```
使用
```

```
286396
```

```
[root@develop1 ~]# umount /mnt
```

Storage Driverごとデータ使用量確認

aufs/btrfs/zfsについては、Dockerファイルシステムディレクトリ以下から使用量を確認する事が出来る

```
root@develop1:/# df --output="used"  
/var/lib/docker/aufs/mnt/dc7620897bf522016e01c062d7d66342472a4cd73793202d95e6c  
ec964e2be17  
Used  
1863008
```

監視するにはコンテナIDが必要となり、ライフサイクルの早いコンテナ監視を手入力アイテムとして登録していくのは大変

他にも、overlayfsの監視・データコンテナマウント時の使用量チェック等作り込みには時間が必要

その他コンテナAgentインストール時の問題

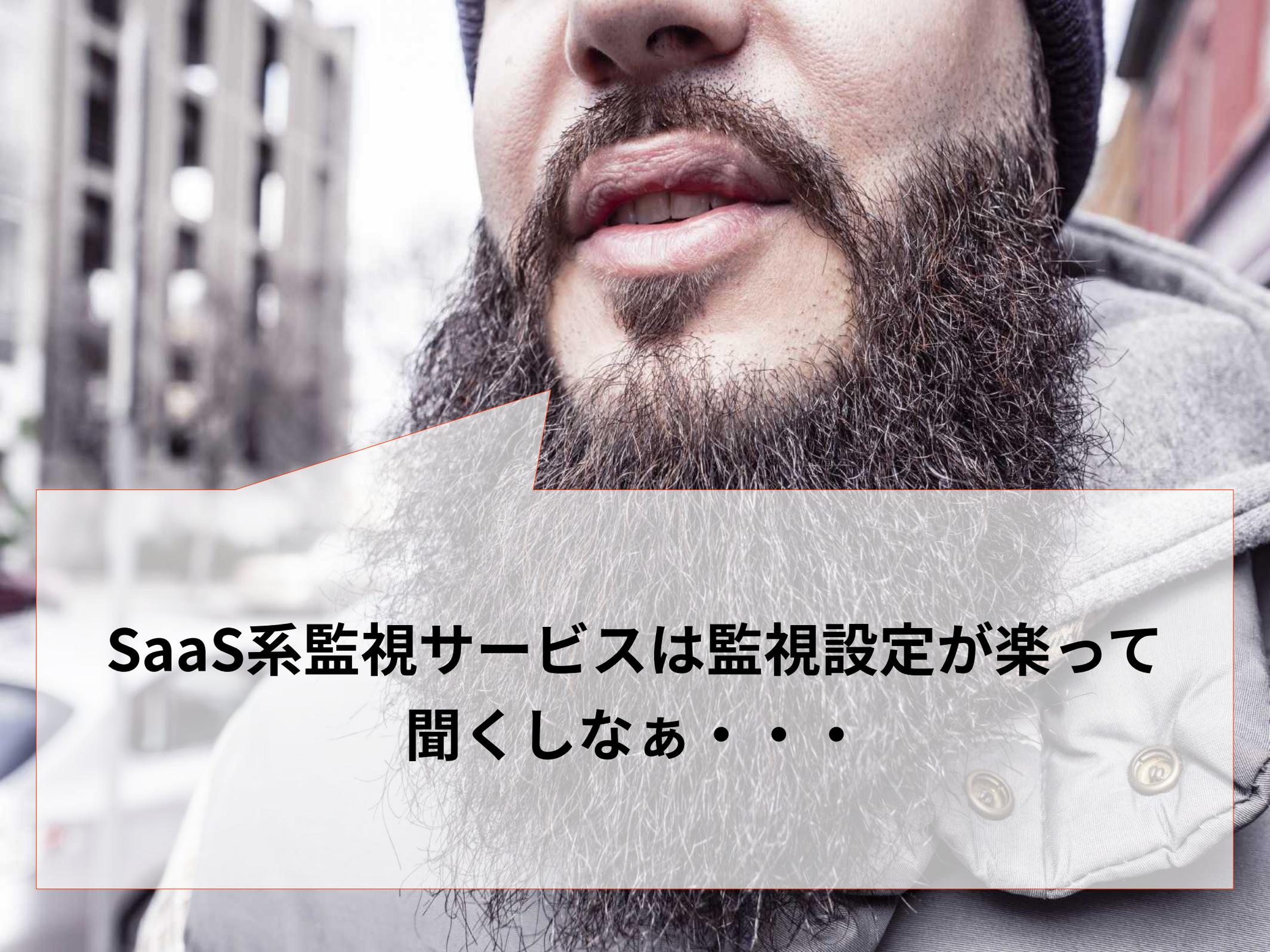


Z

- Dockerホスト監視用のZabbixAgentコンテナを用意しても良いがコンテナ再立ち上げにより変更されるIP (再作成の場合はコンテナのホスト名)も変更されるのでActiveにZabbixサーバへ初期情報を伝える必要がある

→ DockerホストにZabbixAgentをインストールし、統合監視を行うのがbetter.





**SaaS系監視サービスは監視設定が楽って
聞くしなあ・・・**



SaaS監視サービスを使用しますか？

コンテナの監視は少し面倒だし、監視サーバを運用・メンテナンスする費用も決して安くは無いです。

が・・・

- 外部へのデータ保管は許容出来ない**
- 既にZabbixでAgent/SNMP/JMX/VM監視をしているので、一元化したい**
- まだ低予算プロジェクトの為、有償製品は極力使用したくない(´ー`)。○ (将来的には予算つくからサポートも受けたい)**
- これ以上監視用ドキュメント増やしたくない(個人的意見)**



Zabbixでの収集ツールを制作
[3/5]

コンセプト

- コンテナへ監視用Agentをインストールしない
- Dockerホストでコンテナが起動される都度
自動で監視下に置かれる
- ライフサイクルが早いので
コンテナが削除されると監視項目も自動で削除
- ディスク監視については[Maybe in the future :)]
※今後の状況により使用量監視を加える

Zabbix_senderを使用した自動登録・送信

Docker Host

開始

コンテナ一覧
取得

CPU/MEM取得

NW I/O取得

Docker Host事前準備

1. zabbix_sender用意
2. LLD用、JSON形式送信スクリプト用意
3. cron等、ジョブ管理ツールでスクリプト実行

コンテナ登録(LLD)とコンテナ監視データ送信
[zabbix_sender]

Z

Zabbix_senderを使用した自動登録・送信

Docker Host

開始

コンテナ一覧
取得

CPU/MEM取得

NW I/O取得

Zabbix Serverへの事前準備

1. Docker監視テンプレートをインポート
2. Dockerホストへテンプレートの割当

テンプレートの中身

- Trapperで待ち受けるLLD
- LLDの取得結果表示スクリーン

コンテナ登録(LLD)とコンテナ監視データ送信
[zabbix_sender]

Z

Zabbix_senderを使用した自動登録・送信

Docker Host

開始

コンテナ一覧
取得

CPU/MEM取得

NW I/O取得

Doker Host取得データを
zabbix_senderにて送信

dockerステータスディレクトリから
コンテナ一覧を取得

Default: /var/lib/docker/containers

取得したいデータを収集

/sys/fs/cgroup/[memory,cpu,cpuacct]/docker-{コンテナID}.scope

json形式でデータの送信

コンテナ登録(LLD)とコンテナ監視データ送信
[zabbix_sender/TCP]

Z

コンセプトモデルの作成

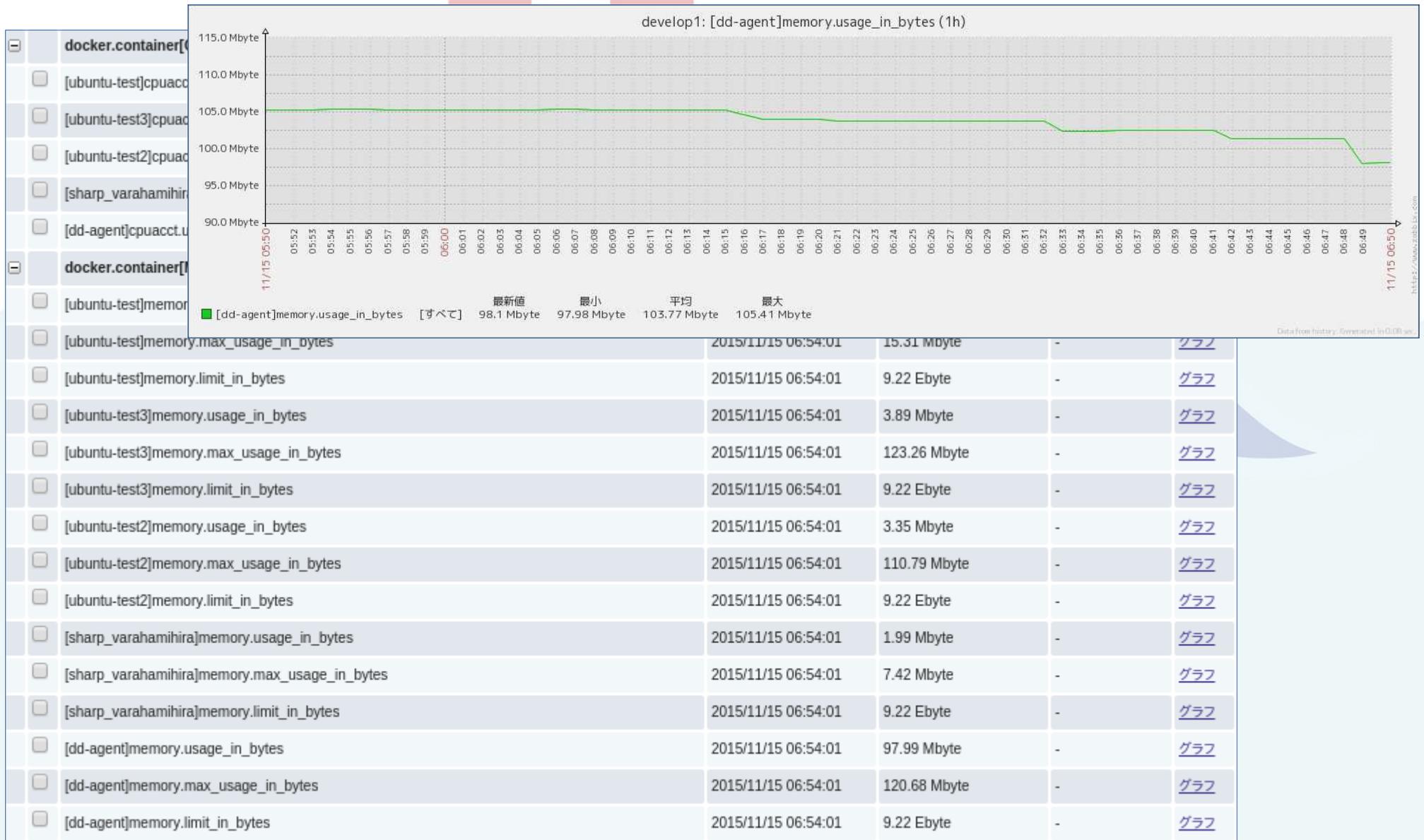
```
#!/bin/env python
# -*- coding: utf-8 -*-

import time,os,json,commands

serverip = '127.0.0.1'
hostname = 'develop1'
keyname = 'vfs.container.discovery'
docker_data_path = '/var/lib/docker/'

def get_container_list():
    container_list = {}
    container_id_list = os.listdir('%scontainers/' % docker_data_path)
    for container_id in container_id_list:
        dict = {}
        file = open(('%scontainers/%s/config.json' % (docker_data_path,container_id)), 'r')
        container_data = json.load(file)
        container_name = container_data.get('Name').lstrip('/')
        container_hostname = container_data.get('Config')['Hostname']
        ...
```

極めて順調ですよ



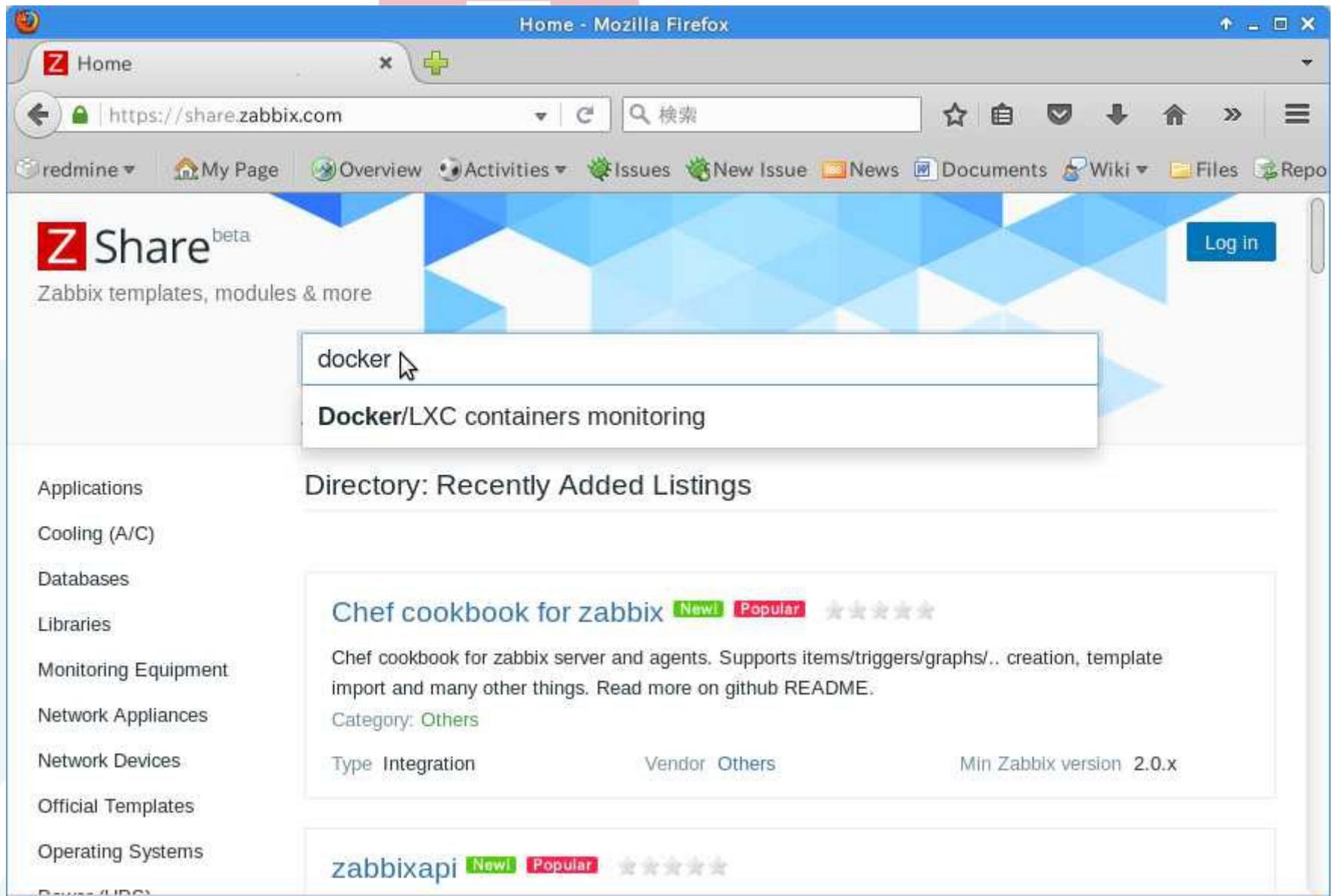
A man with dark hair, wearing a blue sweater, is sitting at a desk in an office. He is looking at a computer monitor on the left, which is partially visible. He has a stressed expression, with his right hand resting against his forehead and eyes. The background is a blurred office setting with a window. There are three text boxes overlaid on the image: one at the top left, one in the center, and one at the bottom right. Lines connect the top-left box to the man's head and the bottom-right box to his hand.

本格的な体制

本格的に作りこもうとしたある日...
[4/5]

作りこむという姿勢

ふとZabbix Shareでdockerと入力してみると



<https://share.zabbix.com/>

Zabbix Share - Docker/LXC containers monitoring - Mozilla Firefox

https://share.zabbix.com/virtualization/doc/ 検索

redmine My Page Overview Activities Issues New Issue News Documents Wiki Files Repo

Applications

Cooling (A/C)

Databases

Libraries

Monitoring Equipment

Network Appliances

Network Devices

Official Templates

Operating Systems

Power (UPS)

Printers

Server Hardware

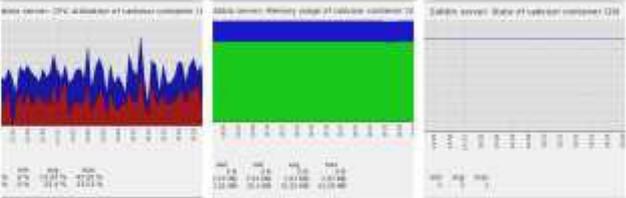
Storage Devices

Telephony

Virtualization

KVM

Docker/LXC containers monitoring Popular



Monitoring of Docker containers (LXC/systemd Docker supported) - Zabbix template and Zabbix C module.
Available CPU, mem, blkio container metrics and some containers config details e.g. IP, name, ... Zabbix Docker module has native support for Docker containers (Systemd included) and should support also a few other container type (e.g. LXC) out of the box.

★★★★★ 2 votes Rating

Listing Details

Type	Template
Vendor	Others
Min Zabbix version	2.2.x



Popular



jangeraj Update README.md		Latest commit 3096db7 14 days ago
doc	mem graph updated	8 months ago
dockerfiles	workdir added - fix #24	2 months ago
src/modules/zabbix_module_docker	Fix memory leakages	3 months ago
template	Cleaning	3 months ago
LICENSE	Initial commit	a year ago
README.md	Update README.md	14 days ago

性能的に負けてる

README.md

Zabbix Docker Monitoring

If you like or use this project, please provide feedback to author - Star it ★.

Monitoring of Docker container by using Zabbix. Available CPU, mem, blkio container metrics and some containers config details e.g. IP, name, ... Zabbix Docker module has native support for Docker containers (Systemd included) and should support also a few other container type (e.g. LXC) out of the box. Please feel free to test and provide feedback/open issue. Module is focused on the performance, see section [Module vs. UserParameter script](#).

Pull requests 0

Pulse

Graphs

HTTPS clone URL

https://github.c...

You can clone with HTTPS or Subversion.

Download ZIP



Zabbixでの収集ツールを制作
Zabbix-Docker-Monitoringを使用する

設定内容

1. ZabbixサーバへTemplateの登録

<https://github.com/monitoringartist/Zabbix-Docker-Monitoring/tree/master/template>

以下、2つをダウンロードし、ZabbixServer-WEBの設定からインポート

Zabbix-Template-App-Docker-active.xml (Passive check)

Zabbix-Template-App-Docker.xml (Active check)

インポートするファイル Zabbix-Tem...ocker.xml

ルール	既存の設定を上書	新規作成	存在しない場合に削除
グループ	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ホスト	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
テンプレート	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
テンプレートスクリーン	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
テンプレートのリンク	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
アプリケーション	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
アイテム	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ディスカバリルール	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
トリガー	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
グラフ	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
スクリーン	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
マップ	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
イメージ	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

設定内容

2. Dockerホスト(ZabbixAgent)にモジュール設置・設定

```
zabbix$ mkdir -p /usr/lib/zabbix/modules  
Zabbix$ wget https://drone.io/github.com/jangaraj/Zabbix-Docker-Monitoring/files/zabbix24/src/modules/zabbix_module_docker/zabbix_module_docker.so -P /usr/lib/zabbix/modules  
zabbix$ vi /etc/zabbix/zabbix_agentd.conf
```

...

```
LoadModulePath=/usr/lib/zabbix/modules  
LoadModule=zabbix_module_docker.so
```

...

```
$ sudo service zabbix-agent restart
```



設定内容

3. Docker コンテナの立ち上げ

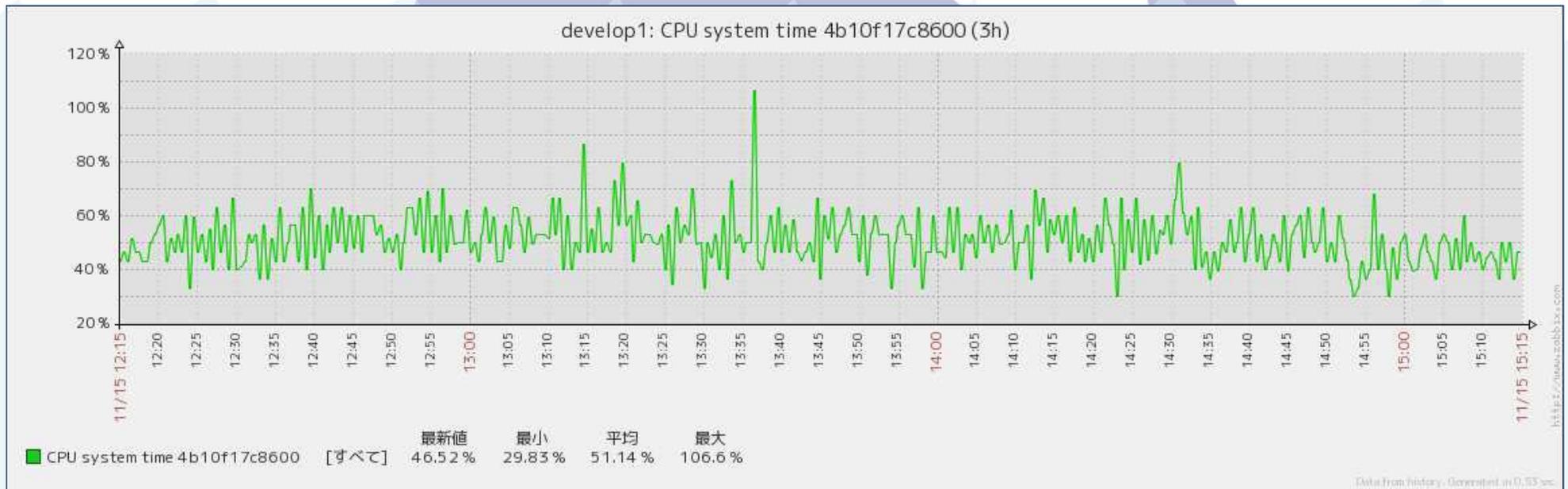
```
$ docker run \  
--volume=/:/rootfs:ro \  
--volume=/var/run:/var/run:rw \  
--volume=/sys:/sys:ro \  
--volume=/var/lib/docker:/var/lib/docker:ro \  
--publish=8080:8080 \  
--detach=true \  
--name=cadvisor \  
google/cadvisor:latest
```

```
$ docker ps | grep cadvisor  
4b10f17c8600    google/cadvisor:latest    "/usr/bin/cadvisor"    7 hours ago    Up 7  
hours         0.0.0.0:8080->8080/tcp    cadvisor
```

設定内容

4. 監視開始

監視はZabbixのLLDにて自動的に監視開始される



設定内容（おまけ）

コンテナ内で出力されるAPログの検知方法の例

例えば、コンテナとして起動されたnginxのログを収集したい場合は

0. コンテナ立ち上げ

```
# docker run --name nginx -d -p 50080:80 nginx
```

1. 標準ストリームを使用し、ログを出力させる

```
# docker exec nginx ln -sf /dev/stdout /var/log/nginx/access.log  
# docker exec nginx ln -sf /dev/stderr /var/log/nginx/error.log
```

2. 外部からwebサーバへ接続を行ってみると・・・

```
# curl -s http://[DockerHost IP]:50080/unknownpage | head
```



設定内容（おまけ）



3. コンテナから出力されたログは、json形式で吐出される

```
# cat /var/lib/docker/containers/[コンテナID]/[コンテナID]-json.log | tail -2 | jq .
{
  "log": "2015/11/18 11:18:26 [error] 5#5: *20
open() \"/usr/share/nginx/html/unknownpage\" failed (2: No such file or directory),
client: 153.127.194.144, server: localhost, request: \"GET /unknownpage HTTP/1.1\",
host: \"zarya.space:50080\\\"\\n\",
  "stream": "stderr",
  "time": "2015-11-18T11:18:26.667531624Z"
}
{
  "log": "153.127.194.144 - - [18/Nov/2015:11:18:26 +0000] \"GET /unknownpage
HTTP/1.1\" 404 168 \"-\" \"curl/7.29.0\" \"-\"\\n\",
  "stream": "stdout",
  "time": "2015-11-18T11:18:26.668702535Z"
}
```

設定内容（おまけ）

4. ZabbixServerのDocker監視ホストにて、log収集アイテムを追加

※ログディレクトリはコンテナのFull IDが必要(64byte)

また、zabbixがlogを直接読込できるかの権限確認も必要

The screenshot shows the Zabbix Item configuration interface. The '名前' (Name) field is 'container stdlog nginx'. The 'タイプ' (Type) is 'Zabbixエージェント(アクティブ)'. The 'キー' (Key) field contains the path 'log[/var/lib/docker/containers/3ec7b0efd3e44f801c91fa8f76d33a4009a' with a '選択' (Select) button. The 'データ型' (Data type) is 'ログ'. The '更新間隔(秒)' (Update interval) is 5, and '履歴保存期間(日)' (History retention) is 90. The 'アプリケーションの作成' (Application creation) section is highlighted in green, and the 'アプリケーション' (Application) dropdown is set to 'Docker'.

```
log[/var/lib/docker/containers/コンテナフルID(3ec7b0efd3e...)/  
コンテナフルID(3ec7b0efd3e...)-json.log,"\"log\": \"(.*)\", \"stream\",,skip,\\1]
```

有効

設定内容（おまけ）

7

5. コンテナのログが収集されます

2015/11/19 11:33:43	-	113.42.5.178 - - [19/Nov/2015:02:33:40 +0000] "GET /test HTTP/1.1" 404 570 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:33:43	-	2015/11/19 02:33:40 [error] 5#5: *32 open() "/usr/share/nginx/html/test" failed (2: No such file or directory), client: 113.42.5.178, server: localhost, request: "GET /test HTTP/1.1", host: "zarya.space:50080"
2015/11/19 11:33:43	-	113.42.5.178 - - [19/Nov/2015:02:33:39 +0000] "GET /test HTTP/1.1" 404 570 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:33:43	-	2015/11/19 02:33:39 [error] 5#5: *32 open() "/usr/share/nginx/html/test" failed (2: No such file or directory), client: 113.42.5.178, server: localhost, request: "GET /test HTTP/1.1", host: "zarya.space:50080"
2015/11/19 11:30:18	-	113.42.5.178 - - [19/Nov/2015:02:30:16 +0000] "GET /test HTTP/1.1" 404 570 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:30:18	-	2015/11/19 02:30:16 [error] 5#5: *31 open() "/usr/share/nginx/html/test" failed (2: No such file or directory), client: 113.42.5.178, server: localhost, request: "GET /test HTTP/1.1", host: "zarya.space:50080"
2015/11/19 11:29:53	-	113.42.5.178 - - [19/Nov/2015:02:29:52 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:29:53	-	113.42.5.178 - - [19/Nov/2015:02:29:50 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:27:13	-	113.42.5.178 - - [19/Nov/2015:02:27:08 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:27:08	-	113.42.5.178 - - [19/Nov/2015:02:27:07 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:19:53	-	113.42.5.178 - - [19/Nov/2015:02:19:51 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:19:53	-	113.42.5.178 - - [19/Nov/2015:02:19:51 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:19:53	-	113.42.5.178 - - [19/Nov/2015:02:19:50 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:19:23	-	113.42.5.178 - - [19/Nov/2015:02:19:21 +0000] "GET /hogehogehoge/ HTTP/1.1" 404 570 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:19:23	-	2015/11/19 02:19:21 [error] 5#5: *29 "/usr/share/nginx/html/hogehogehoge/index.html" is not found (2: No such file or directory), client: 113.42.5.178, server: localhost, request: "GET /hogehogehoge/ HTTP/1.1", host: "zarya.space:50080"
2015/11/19 11:18:03	-	113.42.5.178 - - [19/Nov/2015:02:18:00 +0000] "GET /hogehogehoge/ HTTP/1.1" 404 570 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:18:03	-	2015/11/19 02:18:00 [error] 5#5: *28 "/usr/share/nginx/html/hogehogehoge/index.html" is not found (2: No such file or directory), client: 113.42.5.178, server: localhost, request: "GET /hogehogehoge/ HTTP/1.1", host: "zarya.space:50080"
2015/11/19 11:05:13	-	113.42.5.178 - - [19/Nov/2015:02:05:12 +0000] "GET /hogehogehoge/ HTTP/1.1" 404 570 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/45.0.2454.101 Chrome/45.0.2454.101 Safari/537.36" "127.0.0.1"
2015/11/19 11:05:13	-	2015/11/19 02:05:12 [error] 5#5: *27 "/usr/share/nginx/html/hogehogehoge/index.html" is not found (2: No such file or directory), client: 113.42.5.178, server: localhost, request: "GET /hogehogehoge/ HTTP/1.1", host: "zarya.space:50080"



まとめ [5/5]

- アプリケーションコンテナにZabbixAgentを入れる事はあまり意味が無い。
DockerホストにAgentをインストールして監視を行った方が便利。
※コンテナに付与されるvethもLLDで自動監視される
- ライフサイクルの早いアイテム収集(今回はDockerコンテナ)にはZabbix LLDは非常に効果的
またコンテナ入替えでも、「ディスカバリルール」の「存在しなくなったリソースの保持期間」を設定する事で自動的に監視データを削除する事も可能
- 作る前に類似のテンプレートが存在しないか
<https://share.zabbix.com/> を見よう！

少しだけ宣伝

Tech-Circle - connpas

techcircle.connpass.com

Tech-Circle

@Nishi-Shinjuku

Tech Circle
@Nishi-Shinjuku

Tech-Circle

ITエンジニアのためのインフラ技術と機械学習技術とかの勉強会

現在中イベント [もっと見る](#)

2015/11/16(月) [Tech-Circle Sparkでツイ...](#)

イベント メンバー 資料

B! 0 **G+** 0 **f** いいね! 40 **Twitter** ツイート 7 グループのメンバーです

グループの説明

Tech-Circleとは?

自分が人におすすめしたい技術をシェアし、これまで知らなかった技術に触れる機会を提供する技術勉強会です。

技術に何らか触れる“きっかけ”を作れる場にしたという思いからこの勉強会では講義形式だけではなく、極力参加者自身も勉強会の時間内で技術に触れながら学び、技術を楽しめるようハンズオンの要素を盛り込んでいきたいと考えています。

Tech-Circleは次の2つのテーマのもとに勉強会を開催しています。

- インフラ技術
- 機械学習技術

メンバー (701人)

管理者

他のメンバー

ハンズオン形式で新しい技術に触れる機会を提供しています

インフラ技術をメインでやっていましたが、最近では
機械学習・アプリケーション編も含め月1のペースで
開催しているので、興味がありましたら参加お願いします。

<http://techcircle.connpass.com/>

グループの説明

Tech-Circleとは？

自分が人におすすめしたい技術をシェアし、これまで知らなかった技術に触れる機会を提供する技術勉強会です。

技術に何らか触れる“きっかけ”を作れる場にしたという思いからこの勉強会では講義形式だけではなく、極力参加者自身も勉強会の時間内で技術に触れながら学び、技術を楽しめるようハンズオンの要素を盛り込んでいきたいと考えています。

Tech-Circleは次の2つのテーマのもとに勉強会を開催しています。

- インフラ技術
- 機械学習技術

メンバー (701人)

管理者



他のメンバー







ご清聴ありがとうございました

