OSS監視ツールの失敗事例から学ぶ、知って得する正しい導入手順と注意点

2017/11/29
サイバートラスト株式会社
Linux/OSS 事業部
営業統括部
プロダクトマーケティング部
月城史行

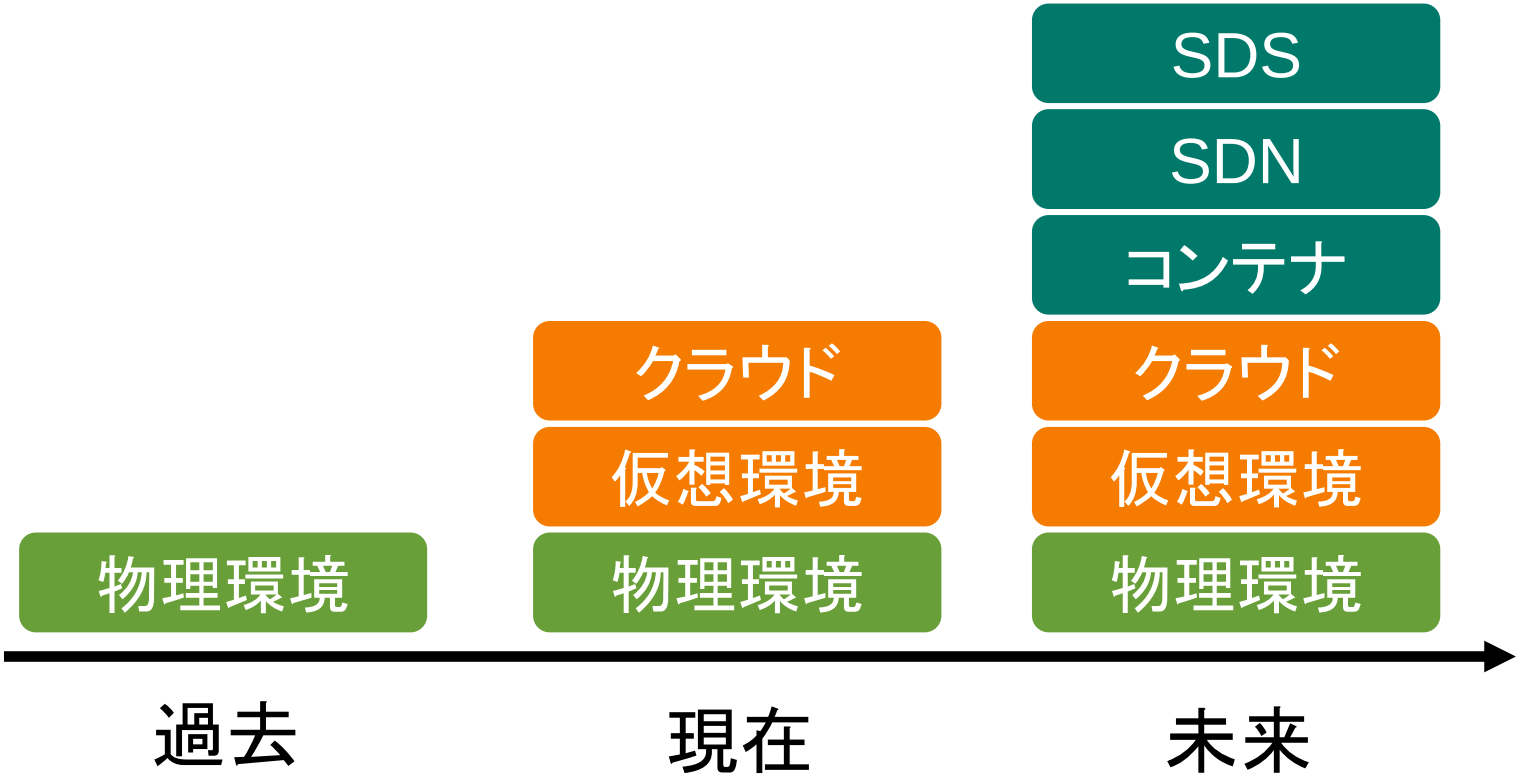
監視の現状と監視設計の大方針

監視の目的

障害検知	システムのダウンタイム最小化
リソース分析	システムのキャパシティプランニング
パフォーマンス分析	システムのサービスレベル評価

システムを安定運用するには監視が不可欠

ITインフラの技術変化

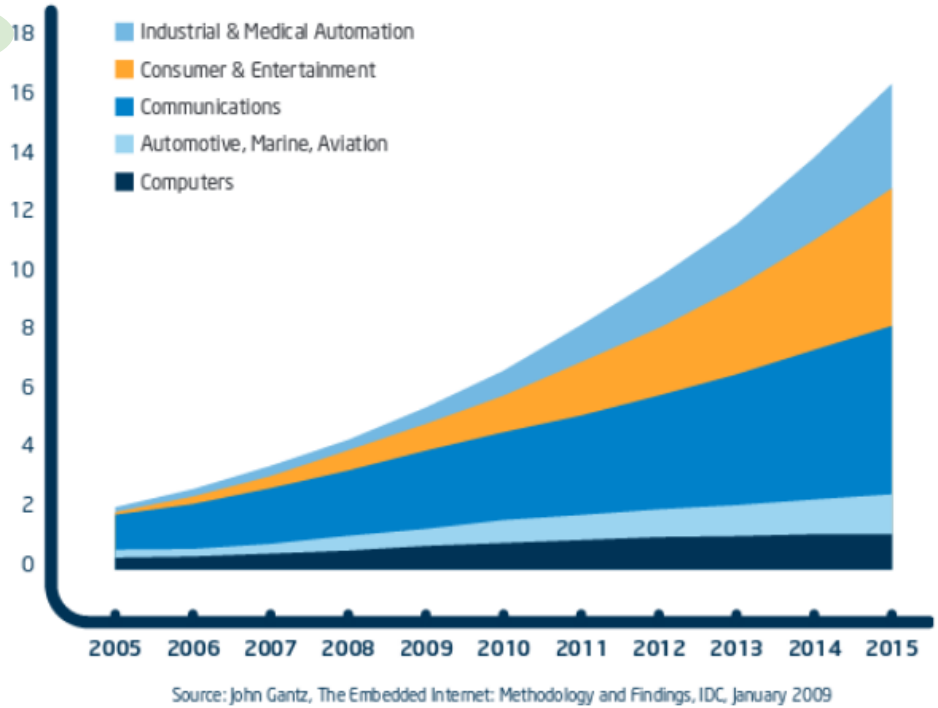


ITインフラの技術要素は増加し、複雑化する一方
ハイパーコンバージド、IoTといった技術要素も登場

監視対象デバイスの増加

2020年には 260億のデバイスがネット接続されると予想

http://www.gartner.com/newsroom/id/2636073



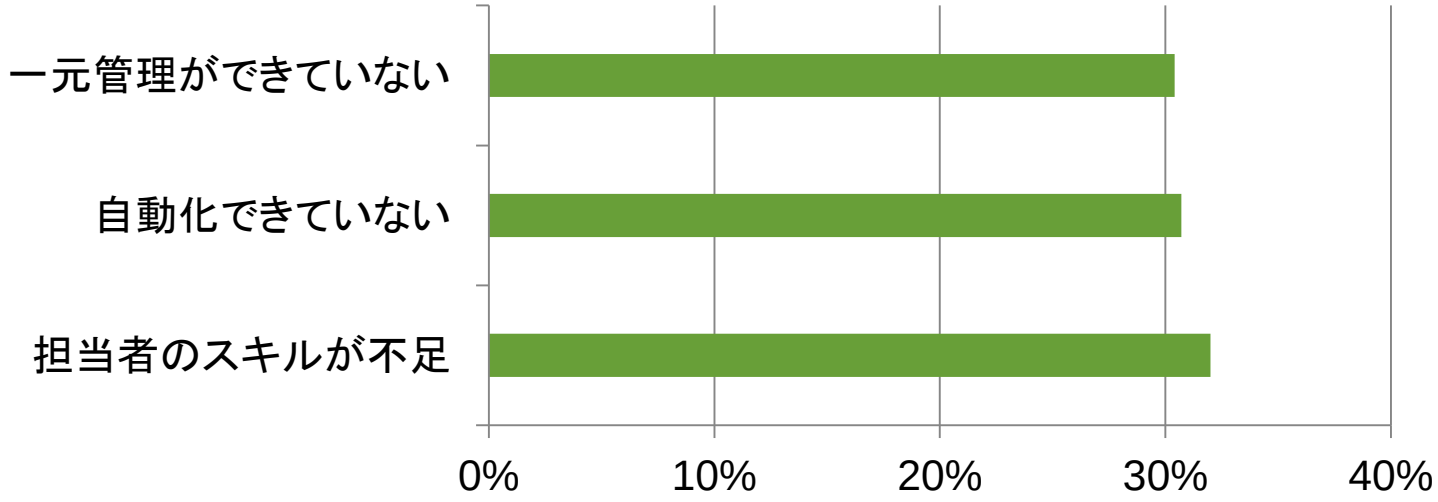
Source: John Gantz, The Embedded Internet: Methodology and Findings, IDC, January 2009

http://download.intel.com/newsroom/kits/embedded/pdfs/ECG_WhitePaper.pdf

日本では2020年の東京オリンピックまでに急激に増加の予想

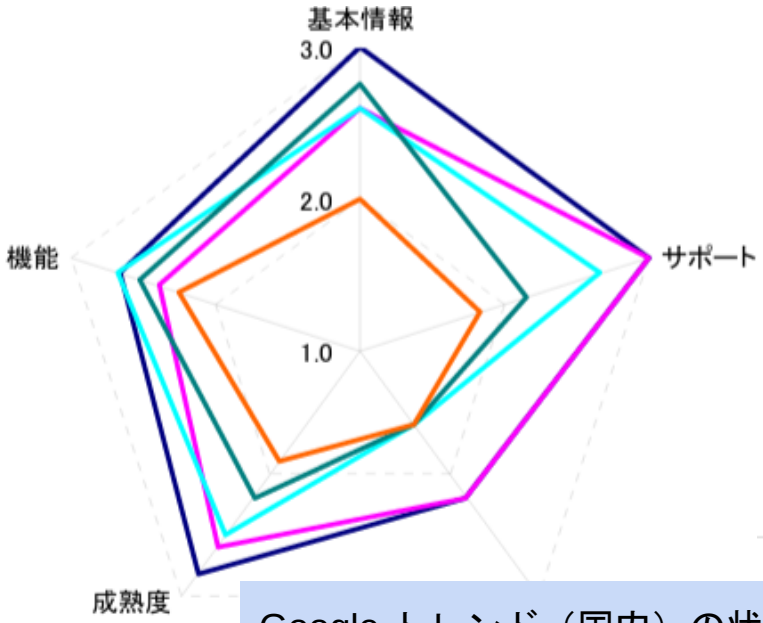
システム運用管理における課題

＊Source: IDCジャパン
国内企業におけるシステム運用管理実態に関するユーザー調査結果
http://www.idcjapan.co.jp/Press/Current/20161020Apr.html



2016年8月、IDCジャパン社が国内企業309社にアンケート。「担当者のスキルが不足」が最多回答となった。過去4回の調査においても、スキル不足が最多回答となっており、改善がすすんでいない。

OSS監視ツールの評価

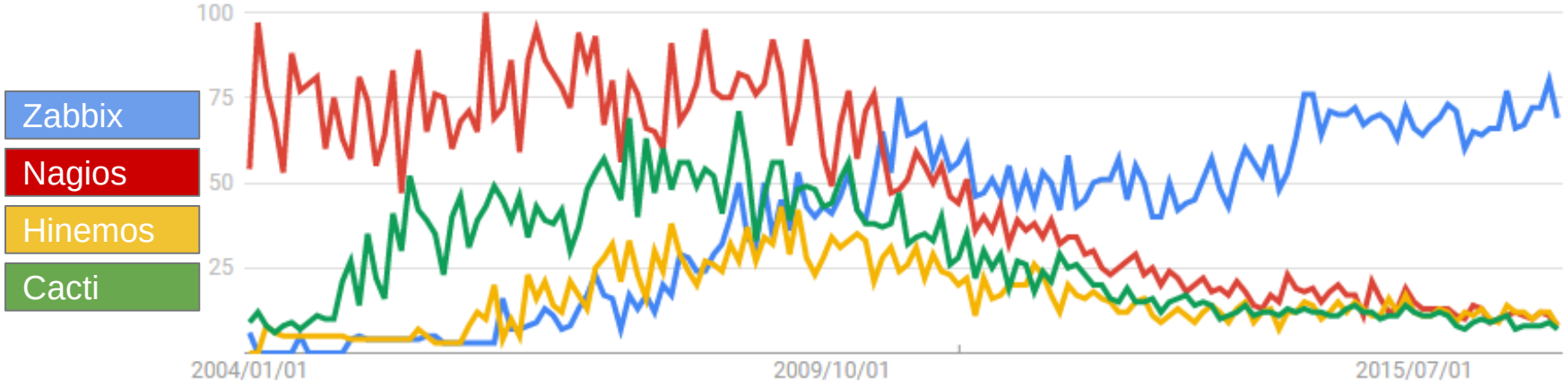


	基本情報	サポート	開発の安定性	成熟度	機能
ZABBIX	★★★★★	★★★★★	★★★★☆	★★★★★	★★★★★
Nagios	★★★★★	★★★★★	★★★★☆	★★★★★	★★★★☆
GroundWork Monitor	★★★★★	★★★★★	★★☆☆☆	★★★★☆	★★★★★
Hinemos	★★★★★	★★★☆☆	★★☆☆☆	★★★★☆	★★★★☆
Xymon	★★★☆☆	★★★☆☆	★★☆☆☆	★★★☆☆	★★★★☆

ZABBIX
Nagios
GroundWork Monitor
Hinemos
Xymon

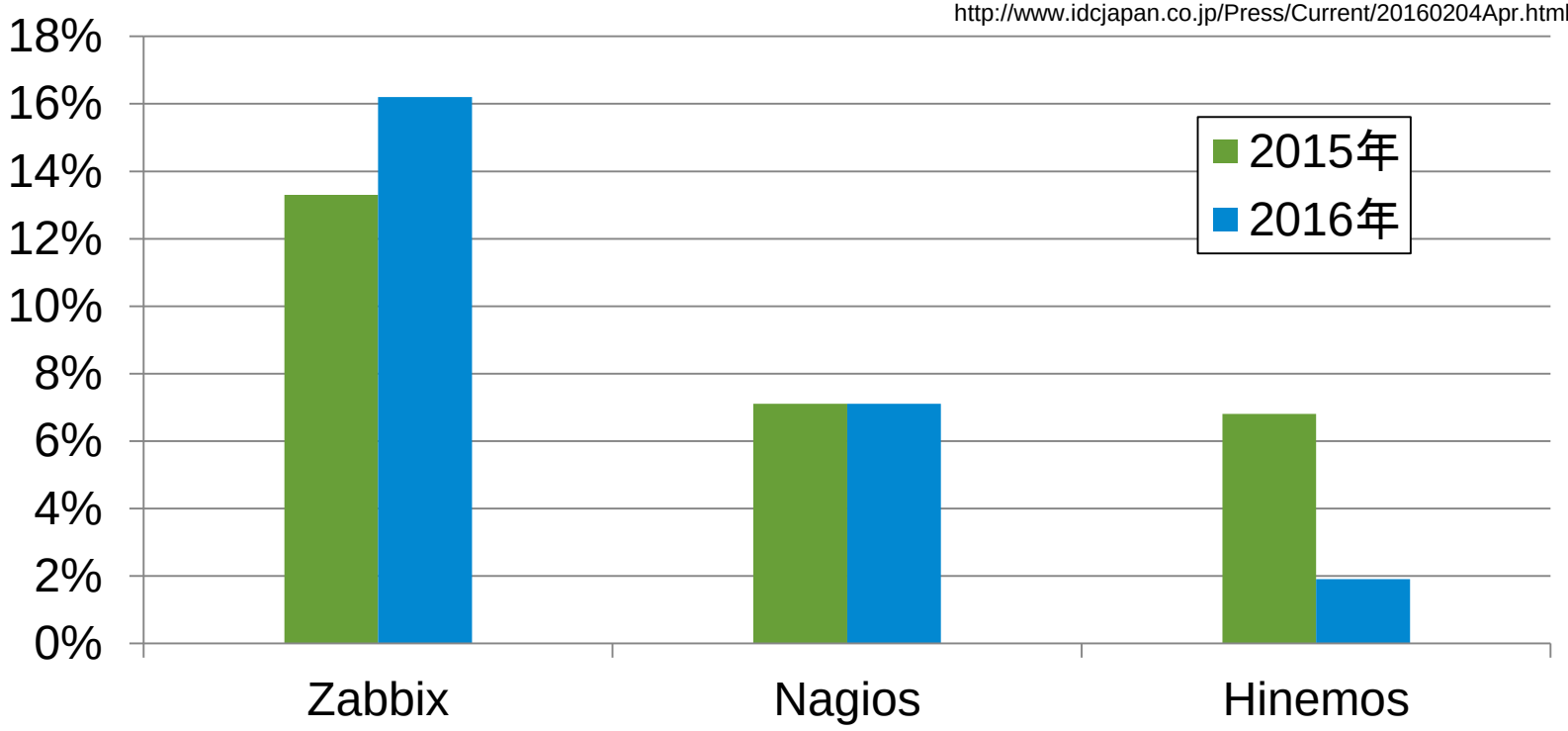
Source: 平成23年6月　独立行政法人　情報処理推進機構
社内向けクラウド構築のために活用できる
ソフトウェアカタログの作成 調査報告書
http://ossipedia.ipa.go.jp/nfs/pdf_pub/1010/209/287/287.pdf

Google トレンド（国内）の状況でも、2010頃からNagiosを逆転して首位



Zabbixのシェア拡大

＊Source: IDCジャパン
2015年 国内オープンソースソフトウェア市場 ユーザー利用実態調査
http://www.idcjapan.co.jp/Press/Current/20150406Apr.html
2016年 国内オープンソースソフトウェア市場 ユーザー利用実態調査
http://www.idcjapan.co.jp/Press/Current/20160204Apr.html



オープンソースソフトウェアのシステム運用管理ソフトは
Zabbixがデファクトスタンダード。

監視設計の大方針

cybertrust

「監視」自体にふりまわされない

監視はシステムを安定運用するための手段の一つ。
監視業務によって、インフラ運用業務に悪影響がでるのは本末転倒。

⇒監視にふりまわされないためには?

監視にふりまわされないために

cybertrust

シンプル

サーバ構成、監視設定は極力シンプルに

- 監視サーバで他のサービスを動かさない
- 複雑な障害判定条件を設定しない
- 障害通知の設定も少なくする

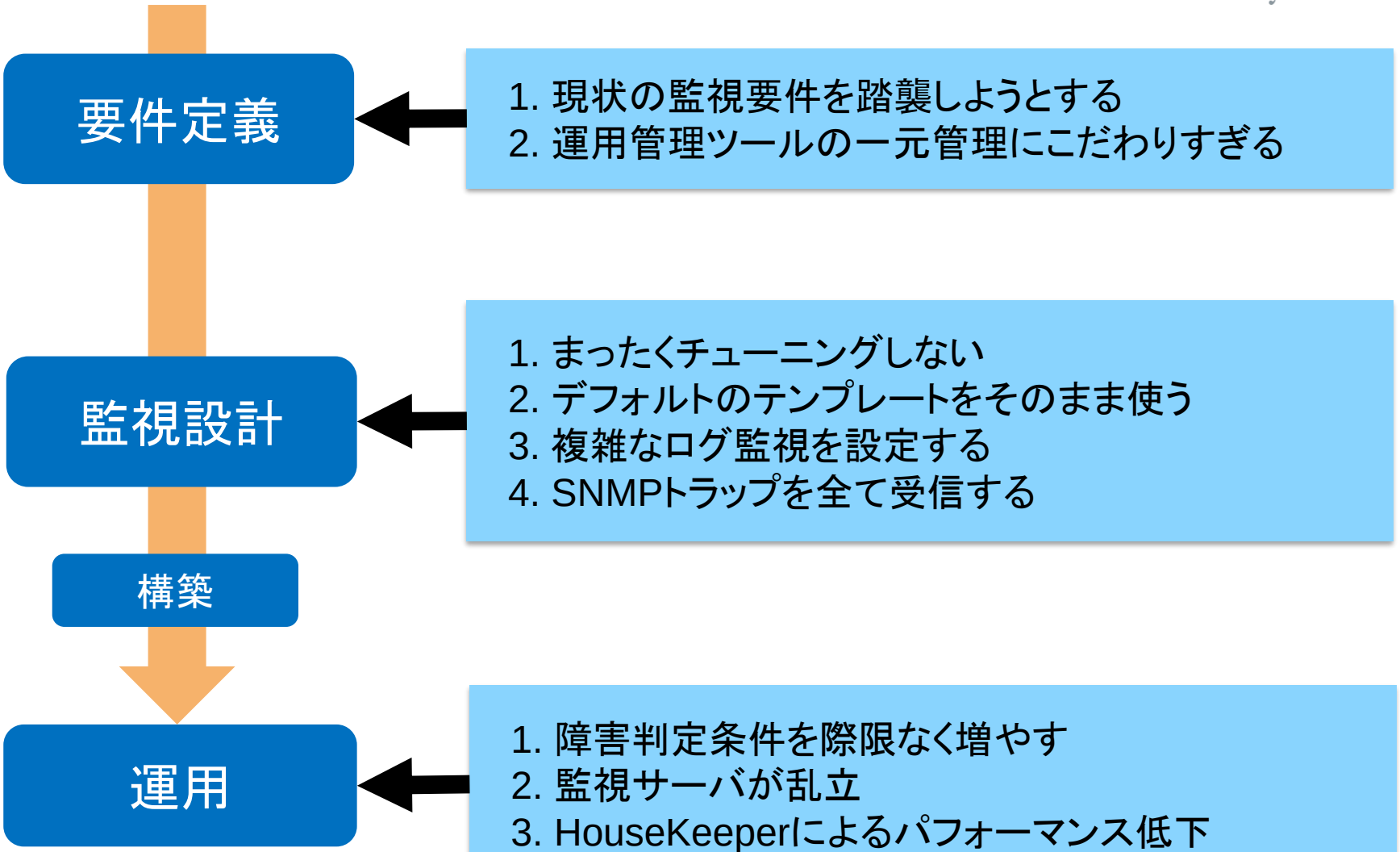
適材適所

監視ツールの得意、苦手な点を把握する
監視機能だけでなく、非機能要件についても考慮が必要

- Zabbixはリソース監視が得意だが、ログやSNMPトラップ監視は苦手
- 障害判定式の「トリガー」、障害通知の「アクション」など用語が独特

よくある落とし穴

よくある落とし穴



要件定義の落とし穴1

cybertrust

■ 現状の監視要件を踏襲しようとする

「監視要件は現状のまま」は要注意。他の監視ツールと思想、設計、実装が異なるため、同じ監視要件をZabbixで実現できないことも。

・**監視、障害通知の除外設定**

「週末のメンテナンス時間中はサービス監視を停止したい」
「ログに特定のエラーコードが出力されると、対象外としたい」
除外設定は他の監視ツールと差異が大きく、問題になりやすい。

・**監視異常時の挙動**

Zabbixは監視失敗時のリトライ回数の調節はできない。
他の監視ツールでは正常と判断されていた監視対象がZabbixにおきかえると障害と判断されることも。
異常時の挙動は要件定義から漏れやすく、運用後に気付く場合が多い。

要件定義の落とし穴2

cybertrust

■ 運用管理ツールの一元管理にこだわりすぎる

リプレース時、運用管理ツールの統合を検討するケースが多い。
既存の運用管理ツールをZabbix一本に統合しようとすると苦戦。

機能	Zabbixの適性	代替ツール
リソース監視	◎	
死活、SNMP監視	◎	
ログ、SNMPトラップ監視	△ 大量の監視、複雑な設定は困難	Fluentdなど
ハードウェア監視	△ SNMP、IPMIで代用	HWベンダーの 監視ツール
パフォーマンス分析 (Web、データベース)	△ 基本的な監視のみ	対象アプリに特化した ツール(APMなど)
インシデント管理	△ イベントでのコメント機能のみ	Redmineなど
ジョブ管理	×	JP1など

監視設計の落とし穴1

cybertrust

■ まったくチューニングしない

Mariadbのデフォルト設定ではパフォーマンスが出にくい。
バッファプールサイズを増やすのは非常に有効。(デフォルトは128MB)

・**Mariadb**

```
innodb_buffer_pool_size = 物理メモリの5〜8割
innodb_log_file_size  = 256MB
innodb_log_files_in_group = 2
```

物理メモリが少ない場合、パラメータが以下の式となるように調整
innodb_log_file_size * innodb_log_files_in_group < innodb_buffer_pool_size

```
innodb_file_per_table = 1
innodb_file_format = Barracuda
```

CPUに余裕があり、フラッシュストレージを使用している場合
DBテーブル圧縮も検討。書き込みサイズの減少により高速化だけでなく、
フラッシュの長寿命化も期待できる。

監視設計の落とし穴1(続き)

cybertrust

■ まったくチューニングしない

監視対象が多い場合、Zabbixもデフォルト設定ではパフォーマンス不足に

・**Zabbix**

ZabbixはPollerと呼ばれるプロセスを複数立ち上げて、監視を行っている。
設定ファイル(zabbix_server.conf)でPollerの設定を行う。

```
StartPollers =
StartTrappers =
StartPingers =
StartHTTPPollers =
```

Zabbixインターナルアイテムで負荷状況を見ながらPollerの数を増やす。
デフォルトで設定されてある「Template App Zabbix Server」に
パフォーマンス関連の監視アイテムが定義されてある。

監視設計の落とし穴2

cybertrust

■ デフォルトのテンプレートをそのまま使う

Zabbixではデフォルトの監視テンプレートがOS別に定義されている。
リソース監視以外の監視アイテムも含まれているので、再設定を検討。

・**パフォーマンス以外の監視アイテム**

/etc/passwdのチェックサム、ホスト名、システムのuptimeなど
不要なら削除

・**agent.ping**

Zabbixエージェントの死活判定にはシンプルチェックを推奨
詳細は弊社のブログにて公開中
https://www.miraclelinux.com/tech-blog/835fku

・**監視間隔**

デフォルトではCPU、メモリなどは1分間隔で監視
重要度が低ければ、5分や10分間隔に変更

監視設計の落とし穴3

cybertrust

■ 複雑なログ監視を設定する

Zabbixの仕様上、検知したくない文字列を設定するのは困難。

⇒他の監視アイテムを検討する

・ログ監視の代わりにプロセス監視、サービス監視を使う
・ログを監視するコマンド(スクリプト) をZabbixエージェントのユーザパラメータから実行する
・定期的にログ監視プログラムを実行し、異常発見時にはzabbix_senderでZabbixサーバに通知する

⇒正規表現は多用しない

正規表現を駆使した設定は、可読性も保守性も低い。弊社のサポートでも正規表現に関する問い合わせは多く、設定ミスの温床となっている。

監視設計の落とし穴4

cybertrust

■ SNMPトラップを全て受信する

SNMPトラップ監視はパフォーマンスが悪く、柔軟な設定ができない。

⇒SNMPトラップの数を減らす

・緊急度の高いSNMPトラップのみ送出し、他のトラップは止める

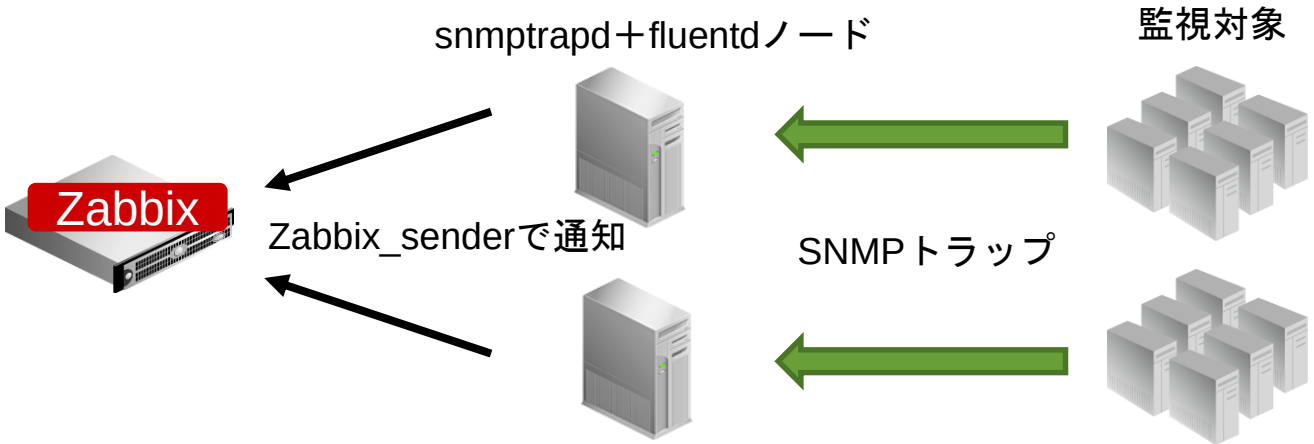
・SNMPポーリングで監視できる機器は、SNMPポーリング監視に統一

⇒複数のsnmptrapdノードとfluentdで負荷分散させる

SNMPトラップをうけるsnmptrapdノードとZabbixサーバを分離

snmptrapdノードにはログ処理用のfluentdと通知用のzabbix_senderをセット

弊社のテックブログにて手法を公開中。



運用の落とし穴1

cybertrust

■ 障害判定条件を際限なく増やす

誤検知、新たな障害が見つかるたびに障害判定条件を追加。
条件を増やすことで、複雑化して運用負荷が増える。

⇒障害への根本対応

CPUやディスク等のリソース不足は、リソースを増強して余裕を持たせる。
アプリの過負荷の場合、アプリの調査を依頼する。

⇒しきい値の変更

障害対応後に問題がない事を確認できれば、障害判定のしきい値を変更する。
これによって、アラートを減らす。

⇒条件を追加する場合、同時に削除できる条件を探す

1つ条件を追加するのであれば、1つ条件を削除するのを目標とする。
あるいは、既存の条件の変更で対応できないか検討する。

運用の落とし穴2

cybertrust

■ 監視サーバが乱立

大企業やITベンチャーで、システムや組織ごとにZabbixサーバが独自に構築、運用されるパターンが散見。

・担当者が不在になり、メンテナンスできなくなる

Zabbixサーバを構築、設定した担当者が退職、異動。組織内で分かるメンバーが不在になり、運用できなくなる。Zabbixは設定の自由度が高いので、複雑になりがち。

⇒動作環境、監視設定の標準化

ZabbixサーバのOS、データベースやバージョンの方針を決める。
Zabbixも最新機能が必須でない限り、LTS版を使用する。

監視設定も個々の監視アイテムではなく、全体的な方針を決める。
「外部スクリプトは使用しない」「ネットワーク機器は死活監視のみ」など

運用の落とし穴3

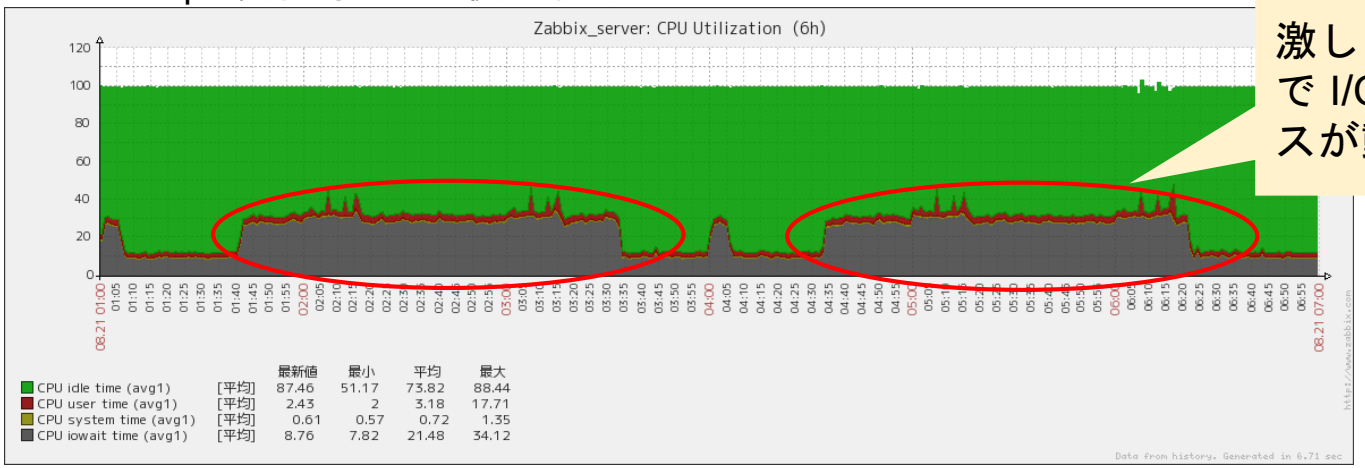
cybertrust

■ HouseKeeperによるパフォーマンス低下

保存期限を過ぎた監視データはHouseKeeperによってデータベースから削除される。長期間運用され、削除対象データが増えるとHouseKeeperの負荷が増大。監視遅延などの問題が発生する。

Zabbixのバージョンアップごとに改良が加えられ、新しいバージョンでは深刻ではなくなった。

HouseKeeper実行時のCPU使用率（4core）



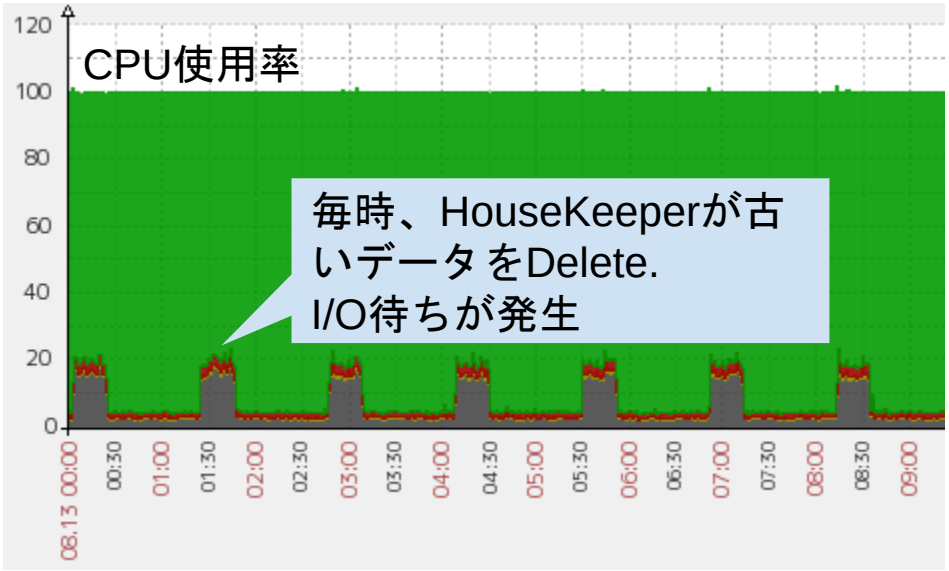
激しい書き込み処理で I/O待ちでプロセスが動けない状態

運用の落とし穴3(続き)

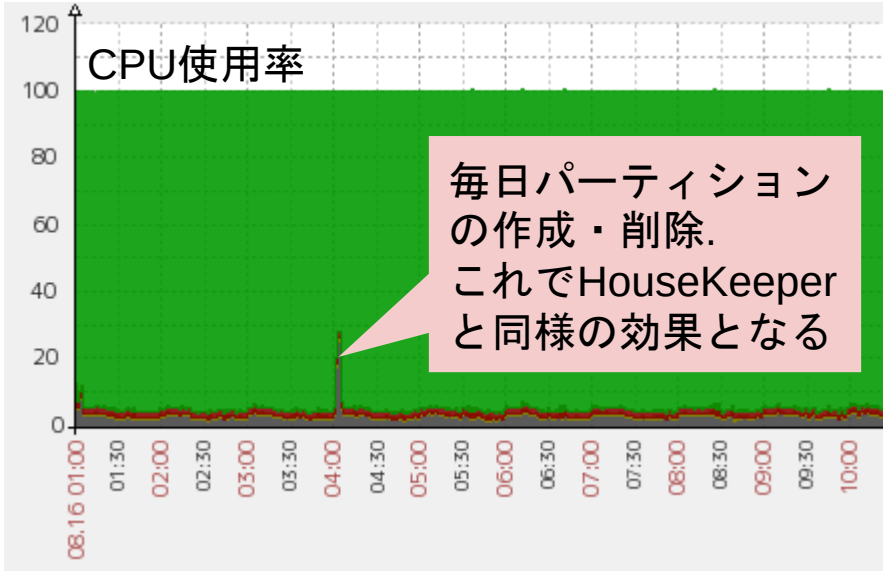
■ HouseKeeperによるパフォーマンス低下

⇒テーブルパーティショニングによる負荷軽減
HouseKeeperを停止させ、DBテーブルをパーティショニングする。
過去データの削除は、パーティションファイルを削除で実現。

OSS版Zabbix 2.0
MySQL5.1
DBパーティショニングなし



MIRACLE ZBX 2.0
MySQL5.5
DBパーティショニングあり



落とし穴を避けるために(運用前)

cybertrust

- 移行時には監視要件を見直す
  「現状踏襲」は複雑になりやすいのでさける。監視ツールの得意、苦手な点を把握して要件を整理。除外設定や監視異常時の挙動にも隠れた要件があるので要注意。

- シンプル、スモールスタート
  初めから複雑な監視設定で運用しない。最低限の監視設定からスタートして必要に応じて監視設定を増やしていく。
  ただし、最低限の設定であってもZabbix、データベースはチューニングを推奨。

- Zabbixと他のツールとの連携
  Zabbixが苦手とする分野は他のツールとの連携で解決できることも。
  当然、運用管理ツールが増えることによる管理コスト増大とのトレードオフとなる。

落とし穴を避けるために(運用後)

cybertrust

- 障害判定条件の追加は慎重に
  障害判定条件を増やす前に、障害の根本対応やしきい値の変更ができないか検討する。それでも障害判定条件を追加する場合、削除できる条件がないか十分に検討する。

- 監視サーバの乱立
  商用ソフトと違いOSSは自由に使うことができるので、導入の敷居が低い。
  単なる実験用の監視サーバが知らぬ間に、本番環境の監視をしている事も。
  担当者が不在になると、運用が行き詰るので、早めに標準ルールを決めて、社内に浸透させる。

- Housekeeperによるパフォーマンス低下
  大規模監視環境で古いバージョンのZabbixを使っている場合は要注意。
  新しいバージョンのZabbixを使う、もしくはテーブルパーティショニングを検討する。

落とし穴を避ける便利な小技

他のツールとの連携用のコマンド

cybertrust

■ zabbix_get

Zabbixエージェントと通信して、情報を取得するツール。
動作確認、監視障害時の問題切り分けに利用できる。
$ zabbix_get –s [IPアドレス] –p [ポート番号] –k [アイテムキー]

```
$ zabbix_get –s 127.0.0.1 –k system.cpu.num
   2   ←CPU数が返ってくる
```

Zabbix 3.0以降、ZabbixサーバとZabbixエージェント間の通信暗号化機能が実装されたが、zabbix_getもオプションで暗号通信可能。

■ zabbix_sender

Zabbixサーバと通信して、監視情報を送信するツール。
事前にZabbixサーバで監視アイテムの設定が必要。
$ zabbix_sender –s [IPアドレス] –s [ホスト名] –k [アイテムキー] –o 値

チューニング時に役立つ監視アイテム

cybertrust

- 「Template App Zabbix Server」の監視アイテム

監視アイテム名	Zabbix_server.conf
Zabbix busy discoverer processes, in %	StartDiscoverers
Zabbix busy escalator processes, in %	StartEscalators
Zabbix busy history syncer processes, in %	StartDBSyncers
Zabbix busy http poller processes, in %	StartHTTPPollers
Zabbix busy icmp pinger processes, in %	StartPingers
Zabbix busy ipmi poller processes, in %	StartIPMIPollers
Zabbix busy java poller processes, in %	StartJavaPollers
Zabbix busy poller processes, in %	StartPollers
Zabbix busy snmp trapper processes, in %	StartSNMPTrapper
Zabbix busy trapper processes, in %	StartTrappers

基本的には、監視アイテムと設定ファイルの設定項目は同じ。
例外はhistory syncerとDBsyncers。

技術情報を掲載開始！！（テック・ラウンジ）

MIRACLE ZBX や Zabbix の構築・設定、内部動作など、技術的なTipsを紹介するポータルを作りました。
ぜひご覧ください。

https://www.miraclelinux.com/product-service/zabbix/tech-lounge

定期セミナーのお知らせ

サイバートラストでは3カ月に一度、東京と大阪でセミナーを実施しています。興味のある方は、ぜひご参加ください。

■ 内容
東京はサポート事例や最新版の新機能を中心とした技術的内容。
大阪は導入や運用のヒントなど。

■ 次回
2018年2月予定

■ 場所
東京: 東新宿、大阪: 梅田

■ URL
https://www.miraclelinux.com/event-seminar/schedule

cybertrust

信頼とともに

ソフトバンク・テクノロジー グループ

ソフトバンク・テクノロジー　エムソリューションズ　フォントワークス　環　サイバートラスト　アソラテック　リデン　モードツー