# SuperKEKB における Zabbix を用いた監視システムの構築

# MONITORING SYSTEM WITH ZABBIX AT SUPERKEKB

佐々木信哉#, A), 中村達郎 A), 廣瀬雅哉 B)
Shinya Sasaki #, A), Tatsuro Nakamura A), Masaya Hirose B)
A) High Energy Accelerator Research Organization (KEK)
B) Kanto Information Service Co., Ltd.

*Abstract*

Monitoring IT infrastructure is important to operate accelerator stably. We had a difficulty to monitor more than 100 instruments with Cacti. In order to monitor IT infrastructure efficiently, we deployed the monitoring system with Zabbix. The system issues an alert by e-mail when a system problem is detected. The archived data in Zabbix are visualized on Grafana, which allows us to easily create dashboards and analyze the data. In addition, we developed a Channel Access client application that sends PV values to Zabbix server and the status of each IOC is monitored with it.

## 1. はじめに

加速器制御に用いている計算機やネットワークスイッチなどの IT インフラを監視し、正常な状態を保つことは加速器を安定して運転するために重要である。

SuperKEKB[1]ではこれまで Cacti[2]を利用して IT インフラの監視を行ってきた。しかし、100 台を超える規模の機器を Cacti で監視するのは管理や可視化の面で困難であった。また、Cacti を利用した監視システムではアラートを利用していなかった。そのため、機器に障害が発生していることに気づかず、後になってから Cacti で状況を確認することも多かった。

より効率的な機器の監視を行うために、我々は監視システムに Zabbix[3]を導入した。また、計算機やネットワークスイッチなどの機器だけでなく、EPICS[4]の PV 値を Zabbix で監視するためのシステムを構築した。これにより、EPICS IOC のリソースや Channel Access (CA)の状態を Zabbix 上から監視することが可能となった。

本稿では構築した監視システムの詳細と利用状況に関して報告する。

## 2. Zabbix

Zabbix はオープンソースの監視ソフトウェアツールである。2004 年にバージョン 1.0 がリリースされてから現在まで継続的にアップデートが行われている。監視ツールとして Zabbix を選択したのは以下のような利点のためである。

- 安定版(LTS)は 5 年間サポートされる。
- これまでも継続的にアップデートが行われており、今後も十分にサポートが行われることが期待できる。
- テンプレート機能により、監視項目や障害設定(トリガー)をホストごとに共有することで、機器の統一的な管理ができる。
- ディスカバリ機能により、監視対象となるホストやその監視項目を自動登録することが可能である。

Zabbix では主要なテンプレートが公式に用意されており、SuperKEKB で利用しているテンプレートの多くはそれらをもとに作成している。また、ディスカバリ機能によって Zabbix 導入時の機器の登録を非常に簡単に行うことが出来た。

# shinya.sasaki@kek.jp

## 3. 監視システムの構成

### 3.1 動作環境

Table 1 に Zabbix が動作する計算機の構成を示す。Zabbix は 1 台のホスト上で動作している。Zabbix から利用するデータベースの MySQL も同じホスト上で動作する。また、データの可視化のために利用している Grafana[5]も同じホスト上で動作する。それぞれのアプリケーションを実行させるために仮想マシンやコンテナは使用していない。

Table 1: Hardware Specification of the Server

| | |
|---|---|
| 機種 | Dell PowerEdge R430 |
| CPU | Intel® Xeon® CPU E5-2603 v3 @ 1.60 GHz |
| Memory | 32GB |
| HDD | DELL Storage MD1400<br>6 TB SAS Disk |

利用している Zabbix のバージョンは 4.0 LTS である。Zabbix のバージョンは加速器が停止した際に随時アップデートするようにしている。2019 年 7 月現在、163 ホスト、約 33000 アイテム、約 10000 トリガーの監視を行っている。

### 3.2 アラートの設定

Zabbix は障害(トリガー)発生時にアラートを送信する機能を有している。障害には「情報」・「警告」・「軽度の障害」・「重度の障害」・「致命的な障害」からなる 5 段階の深刻度を設定することが可能となっている。

SuperKEKB では、障害発生時にメールでアラートを送信するようにしている。これにより、システム管理者が

速やかに障害発生を把握することが可能となった。また、障害の深刻度に応じてアラート送信の振る舞いを変更して運用している。これは重度の障害のアラートが軽度の障害のアラートに埋もれるのを防ぐためである。「重度の障害」以上の深刻度の障害に関しては障害発生時に速やかにアラートが送信される。一方、「警告」・「軽度の障害」に設定された障害に関しては、その障害が 24 時間継続した場合にのみアラートを送信するようにしている。「情報」に設定された障害はアラートを送信しない。

障害のログは Zabbix のデータベースに保存されるため、システムのどの部分においてトラブルが多いか、またどこを改善することができるかなどを考える材料にもなる。

### 3.3 Grafana によるメトリクスの可視化

Zabbix で収集したメトリクスは Grafana 上で可視化している。Grafana は、収集されたデータをグラフなどによって可視化するためのオープンソースのツールである。Web ブラウザから Grafana に接続し、ダッシュボードの作成や閲覧を行うことができる。可視化するデータは、InfluxDB や MySQL・Elasticsearch など様々なストレージバックエンドから取得できる。

Grafana はプラグインを利用して機能を拡張することが可能である。Zabbix のデータを Grafana 上で可視化するために、Zabbix plugin for Grafana[6]というプラグインを SuperKEKB では適用している。このプラグインでは Metric processing functions を利用して、データを変換して表示することができる。例えば、メトリクスの移動平均を表示する movingAverage や、複数のメトリクスを最大値や最小値でソートして、その上位いくつかのデータを表示することができる top などの functions を利用することができる。

Zabbix にもダッシュボード機能は存在するが、操作性や視認性の面で Grafana の方が利用しやすいと考えて、我々は Grafana を可視化ツールとして採用している。また、ほかの様々なストレージバックエンドからのデータも表示することができるため、今後の拡張性が高いことも採用した理由の 1 つである。

Grafana は機能の追加やアップデートが活発に行われているため、新しいバージョンがリリースされる毎にアップデートを実施して利用している。

## 4. 計算機の監視

計算機監視のためのデータ収集は、Zabbix エージェントを利用して行っているものと、SNMP を利用して行っているものがある。Zabbix エージェントは、計算資源やアプリケーションの状態を監視するために、監視対象上で動作するアプリケーションである。Zabbix エージェントが収集した情報は Zabbix サーバーに送信される。Table 2 に 2019 年 7 月現在の計算機のデータ収集方法と用途・台数を示す。

Table 2: Current Status of the Computer Monitoring

| データ収集方法 | 用途 | 台数 |
|---|---|---|
| Zabbix エージェント | サーバー | 2 台 |
| SNMP | サーバー | 15 台 |
| SNMP | 加速器運転用端末 | 6 台 |

これまで Cacti によって監視していた既存の計算機に対しては、なるべく手を加えずに監視を行う方針とした。そのため、多くの計算機は SNMP でデータを収集している。Zabbix 導入後に追加された計算機に関しては、Zabbix エージェントを導入して監視を行っている。計算機監視のテンプレートは、公式で提供される「Template OS Linux SNMPv2」・「Template OS Linux」をもとに作成した。

計算機の監視では、主に CPU 使用率やメモリ使用率・ネットワークトラフィックなどの OS のメトリクスを監視している。

Grafana では、Zabbix エージェントによってデータ収集しているものと SNMP によるものとで、それぞれ別のダッシュボード上で可視化している。SNMP でデータ収集を行っている計算機の監視画面を Fig. 1 に示す。


Figure 1: Grafana dashboard for computer performance monitoring. Each metrics is collected by SNMP.

## 5. ネットワークスイッチの監視

ネットワークスイッチは SNMP によってデータを収集している。Table 3 に 2019 年 7 月現在、監視を行っているネットワークスイッチのベンダーと台数を示す。導入した時期や目的に応じてベンダーは異なるが、Zabbix において統一的に監視を行うことができている。

Table 3: Current Status of the Network Switch Monitoring

| ベンダー | 台数 |
|---|---|
| Cisco | 36 台 |
| Buffalo | 33 台 |
| 三菱電線工業 | 12 台 |
| Apresia | 7 台 |

Zabbix 3.4 以降のバージョンでは Cisco などの大手ベンダー向けのテンプレートが公式で提供されている。そのため、比較的簡単にホストと監視項目の登録を行うことが出来た。

ネットワークスイッチでは主にインターフェイスごとの通信量やブロードキャストパケットの通信量・エラーパケット量・破棄したパケット量を監視している。また、可能であればネットワークスイッチ自身の CPU 使用率やメモリ使用率も監視している。

Grafana で可視化したコアスイッチの通信量を Fig. 2 に示す。このパネルでは、対象のネットワークスイッチの中で最も通信量の多い 5 つのポートの通信量を表示している。これにより、通信量が多くなっているポートのみに注目することができるため、問題となりうる通信にも気づきやすくなっている。例えば、Linac と SuperKEKB 間の通信量が約 500 Mbps 程度までに達することがあることや、大容量ファイルのデータ転送が約 1 Gbps 程度の通信量になることが Fig. 2 から分かる。これに対処するため、対象のネットワークスイッチのアップリンクを 1 Gbps から 10 Gbps に変更した他、中長期的にデータ転送の方法について検討するなどしている。

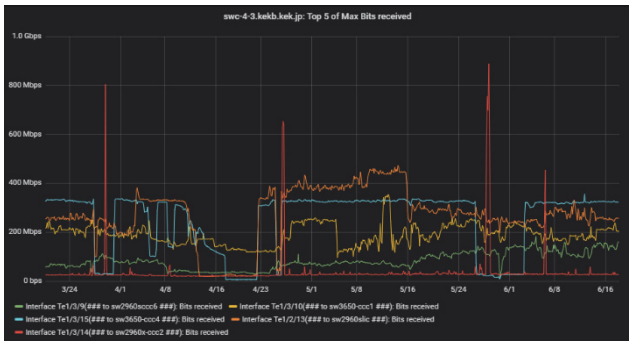


Figure 2: Grafana panel for network traffic. This panel shows received network traffic on top 5 ports in the core switch. Orange line indicates the traffic from Linac is reached to 500 Mbps. Red line indicates large data transfer occupies about 1 Gbps bandwidth.

## 6. EPICS PV 値の監視

### 6.1 背景

SuperKEKB では、加速器運転中に IOC の動作障害が度々発生していた。例えば、CPU 使用率やメモリ使用率が高騰して CA クライアントの新規接続が不可能になることや、IOC に対して CA の大量の subscription 接続が発生して正常に動作することが出来なくなることなどがあった。しかし、IOC の監視はこれまで行っていなかったため、問題に気づいてから状況を確認するこがほとんどであった。そのため、計算機やネットワークスイッチと同様に IOC を監視することが求められた。

IOC の監視は、他の研究施設でも様々な方法が試みられている。

例えば、IOC の死活監視には Alive モジュール[7]や RecSync[8]が利用される。Alive モジュールは、監視サーバーに対して IOC が UDP のハートビートメッセージを定期的に送信して死活監視を行うものである。また、Alive モジュールでは環境変数やシステム情報なども監視サーバーに送信することができる。これに似たシステムである RecSync では、IOC 起動時に自身の持つレコードリストの情報などをサーバーにアップロードした後、サーバーからの UDP の Ping メッセージに対して IOC が Pong メッセージを返すことで死活監視を行う。そのほか、RIBF では ICMP の Ping やポートチェック・caMonitor を利用して IOC のネットワークや Channel Access の死活監視を行っていることが報告されている[9]。

J-PARC では PCMON をベースにしたデバイスサポートを利用して IOC のリソースや状態を PV として取得できるようにしている。そして、その PV を監視する IOC を立ち上げて IOC の監視を行っていることが報告されている[10]。

PF 及び cERL では IOC のタイムスタンプレコードや、ある 2 つの数値レコードの差分を監視する監視用 IOC を用意して IOC の監視を行っていることが報告されている[11]。このシステムでは CSS Alarm と連携してアラートの通知を行うほか、Python で作成したメール通知システムとも連携して動作する。

上記のような監視方法もある中で、我々は IOC の監視に Zabbix が利用できれば、IT インフラと同様に IOC を統一的に監視することが出来るようになると考えた。また、Zabbix の機能を利用することで、上記の監視システムと比較して、障害設定やアラートの設定を柔軟に行うことが出来るようになると考えた。そのため、我々は EPICS の PV 値をメトリクスとして Zabbix に投入するソフトウェアの開発を行った。

Linux 上で動作する IOC の OS メトリクスだけであれば、既存の計算機監視と同様に SNMP や Zabbix エージェントを利用することも可能であった。しかし、VxWorks 上で動作する IOC の OS メトリクスや CA の状態の監視を行うには、EPICS レコードから情報を収集する方法が効率的であると考えて、EPICS PV 値収集ソフトウェアを開発した。

### 6.2 EPICS PV 値収集ソフトウェア

我々は、EPICS PV 値をメトリクスとして Zabbix に送信する CA クライアント zabbix-epics-py[12]を開発した。このソフトウェアでは CA の通信に PyEpics[13]を利用し、Zabbix との通信に py-zabbix[14]を利用する。

使用例を Fig. 3 に示す。zabbix-epics-py では、監視する PV 名、Zabbix に登録済みの監視ホスト名とアイテムキー、そして interval と func を辞書のリストとして登録する。メトリクス登録先となるアイテムのタイプは Zabbix トラッパーを指定して登録する。

```
>>> from zbxepics import ZabbixSenderCA
>>> server_ip = '127.0.0.1'
>>> port = 10051
>>> config = False
>>> items = [dict(host='dummyHost', pv='TEST:PV', interval=30,
                  item_key='zabbix-epics-py-test.item', func='last')]
>>> sender = ZabbixSenderCA(server_ip, port, config, items)
>>> sender.run()
```

Figure 3: Example usage of zabbix-epics-py.

interval にはメトリクスの送信時間間隔を指定する。メトリクスの送信と送信の間、つまり interval の間に更新された PV 値はソフトウェア内部でバッファされる。バッファされた値に対してどのような関数を適用してメトリクスを送信するかは func によって指定する。例えば、last を指定すればバッファ中の最新値を、avg を指定すればバッファ中の値の平均を 1 つのメトリクスとして送信する。func として last・min・max・avg の 4 つが現在用意されている。

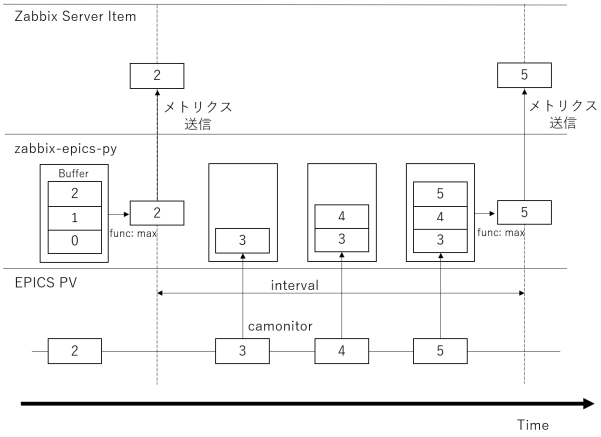


Figure 4: Behavior of zabbix-epics-py. Monitored PV updates 3 times and its values are stored in the zabbix-epics-py buffer. Since func is set as max, 5 which is the max value in the buffer is sent to Zabbix server.

interval に monitor という文字列を指定すれば、更新された PV の値がすべてメトリクスとして送信される。interval が monitor の場合は func の指定は無視される。

Figure 4 に PV 値の更新と interval、func の関係を示す。Figure 4 では interval の間に PV 値が 3、4、5 と更新している。そのため、zabbix-epics-py のバッファには 3 つの値が格納されている。func には max が指定されているため、3 つの値の中の最大値である 5 がメトリクスとして送信される。

SuperKEKB では zetemple[15]というソフトウェアを開発し、その中で zabbix-epics-py をインポートして使用している。zetemple では指定されたホストに登録されてあるテンプレートのアイテムキーをもとに、そのアイテムに送信すべき PV 名を決定する。zetemple の実行の際には Zabbix に登録してある監視ホスト名と、PV のプリフィックスが書かれた csv ファイルを渡して実行する。

### 6.3 IOC 監視への利用

SuperKEKB では、監視対象となる IOC 上で、デバイスサポート devIocStats[16]を動作させている。devIocStats によって IOC の CPU 使用率やメモリ使用率、CA クライアント数などを PV 値として取得できるようになる。これらの PV 値をメトリクスとして Zabbix に送信して監視を行っている。現在は試験的に 32 台の IOC を監視している。interval はすべて 180 秒としており、PV の最新値をメトリクスとして送信するために、func は last を指定している。

devIocStats は 1 秒ごとに 1 ずつカウントアップしていくハートビートレコードを持っている。ハートビートが正常に更新されない場合は IOC が高負荷になっているなどの障害が発生している可能性が高い。そのため、ハートビートを監視することは非常に有用である。SuperKEKB では、ハートビートが 180 秒間で 178 以上増加していない場合を軽度の障害として設定している。また、180 秒間でハートビートの値が変化していない場合は重度の障害として設定している。

そのほか、CA クライアントが大量に接続したことが原因で障害が発生したことがあったことを踏まえて、IOC に対して 100 以上の CA クライアントが接続している場合は警告、200 以上の場合は重度の障害として設定している。

## 7. まとめ

SuperKEKB ではより効率的な機器の監視を行うために Zabbix を用いた監視システムを構築した。Zabbix のアラートや Grafana の可視化を利用することで障害への対応が早くなっただけでなく、これまで気付かなかった問題にも気づくことが出来るようになった。また、テンプレートやディスカバリの機能によって、効率的に機器の登録管理を行うことが可能となった。

計算機やネットワークスイッチなどの IT インフラの監視のほか、EPICS PV 値をメトリクスとして監視できるようにシステムを構築した。これにより、IOC のリソースや CA の状況が Zabbix を通して監視することが可能となった。

## 参考文献

[1] Y. Ohnishi *et al.*, “Accelerator design at SuperKEKB”, Prog.Theor. Exp. Phys. (2013) 03A011; http://ptep.oxfordjournals.org/content/2013/3/03A011.full.pdf
[2] https://www.cacti.net
[3] https://www.zabbix.com
[4] https://epics-controls.org
[5] https://grafana.com
[6] https://grafana.com/plugins/alexanderzobnin-zabbix-app
[7] https://github.com/epics-modules/alive
[8] https://github.com/ChannelFinder/recsync
[9] A. Uchiyama *et al.*, “An Attempt to Implement the Alive Monitoring System for Reliable EPICS-based RIBF Control System”, Proceedings of the 13th Annual Meeting of Particle Accelerator Society of Japan, Chiba, Japan, August 8-10, 2016, pp. 664-667; https://www.pasj.jp/web_publish/pasj2016/proceedings/PDF/MOP1/MOP100.pdf
[10] H. Nemoto *et al.*, “IOC Surveillance System for J-PARC MR Control”, Proceedings of the 9th Annual Meeting of Particle Accelerator Society of Japan, Toyonaka, Japan, August 8-11, 2012, pp. 745-748; https://www.pasj.jp/web_publish/pasj9/proceedings/PDF/WEPS/WEPS118.pdf
[11] Y. Kameta *et al.*, “Development of equipment monitoring system for PF and cERL”, Proceedings of the 15th Annual Meeting of Particle Accelerator Society of Japan, Nagaoka, Japan, August 7-10, 2018, pp. 593-596; https://www.pasj.jp/web_publish/pasj2018/proceedings/PDF/WEP0/WEP095.pdf
[12] https://github.com/sasaki77/zabbix-epics-py
[13] https://cars9.uchicago.edu/software/python/pyepics3
[14] https://github.com/adubkov/py-zabbix
[15] https://github.com/sasaki77/zetemple
[16] http://www.slac.stanford.edu/comp/unix/package/epics/site/devIocStats