FOUNDATIONS OF COMPUTING AND DECISION SCIENCES
Vol. 41 (2016) No. 4
DE GRUYTER OPEN
DOI: 10.1515/fcds-2016-0014
ISSN 0867-6356
e-ISSN 2300-3405

TOOLS FOR DISTRIBUTED SYSTEMS MONITORING

Łukasz KUFEL*

Abstract. The management of distributed systems infrastructure requires dedicated set of tools. The one tool that helps visualize current operational state of all systems and notify when failure occurs is available within monitoring solution. This paper provides an overview of monitoring approaches for gathering data from distributed systems and what are the major factors to consider when choosing a monitoring solution. Finally we discuss the tools currently available on the market.

Keywords: distributed systems, monitoring, monitoring solution, monitoring tools

1. Introduction

Monitoring solution has become an inherent part of distributed systems [2]. The main advantage it provides is early failure identification and notification of support teams through email or text message. Some solutions offer an automated task executions to fix the issue immediately hence minimize systems downtime and human intervention. Those systems are constantly ensuring availability of crucial IT infrastructure components, for example Zabbix monitors the system that controls who enters The European Organization for Nuclear Research (CERN) [4] or Ganglia monitors free-content Internet encyclopedia Wikipedia [28].

Designing monitoring solution for distributed systems is a process of understanding how the solution collects and presents the data, what events are available, what can be monitored and measured and eventually a selection of the tool. This article provides guidelines for selecting monitoring solution as well as a review of currently available monitoring tools. There are multiple monitoring tools available on the market [5]. Choosing the right one can be difficult as there are multiple factors to consider. For example free of charge solutions have limited monitoring features and their user interface is usually very simple, and sometimes hard to navigate.

The paper is organized as follows. Section 2 presents basic concepts of monitoring distributed systems. In Section 3 we discuss major areas where monitoring solution brings

* Institute of Computing Science, Poznan University of Technology, Poznan, Poland; Technical Operations Manager, Expedia.com, lukasz.kufel@hotmail.com

benefits to the organization. In Section 4 we describe four approaches how collection of monitoring data can be achieved. Guidelines about selecting the monitoring solution are presented in Section 5. Research results of 15 monitoring tools are provided in Section 6. Sample deployment and monitoring experiment with three popular monitoring solutions is discussed in Section 7. Section 8 contains summary and conclusions.

2. Monitoring Fundamentals

A monitoring solution for distributed systems is composed from various elements such as monitoring layers, events, thresholds, polling intervals and data retention. The layers represent where the data are collected from, how it is transported to the monitoring system and eventually where the results are presented. Once the data are stored in monitoring system as an event, the system depending on threshold configuration can trigger an alert notification to the relevant support team. Monitoring system also requires a policy, which defines how often the data need to be collected.

2.1. Layers

Monitoring layers are sections in solutions design to visualize data flow from monitored systems to monitoring system.

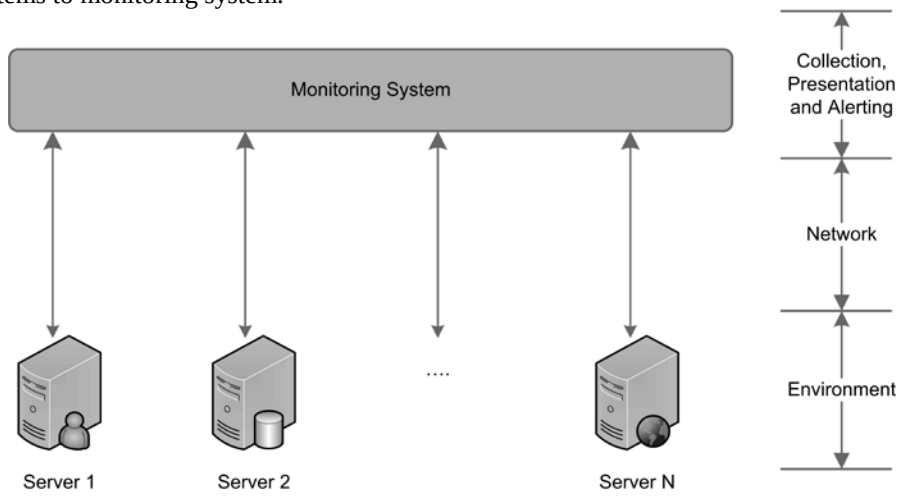

Figure 1. Three major monitoring layers representing sample design of monitoring solution.

Figure 1 shows sample design of monitoring solution which consists of following layers:

- *Environment*. This layer demonstrates distributed systems infrastructure that includes various types of hardware, operating systems, applications and services. It also includes storage devices and databases.
- *Network.* This section of data flow diagram is responsible for transmitting monitoring data from a source (monitored system) to a destination (monitoring system). Network can be seen as a local (Local Area Network) or geographically distributed (Wide Area Network).
- *Collection, Presentation and Alerting*. The core layer in monitoring solution that initiates monitoring, stores collected data, visualize metrics and triggers alert notifications when needed. This is the section where end user configures and controls entire solution as well as runs reports for availability and capacity metrics.

2.2. Events and Thresholds

We define the event similarly to the definition presented by Terenziani et al. [12] and Tierney et al. [13] as time-stamped information about the state of a server or an application and its relevant system metrics, such as availability, CPU utilization, disk utilization and security auditing to industry standards. The contents of every event should be characterized by the meaningful details, such as host name, source name, event type (on Windows) or event severity (in Unix environments), event id and the event message. Table 1 presents five types for events logged on Windows platforms and eight severities on Unix platforms [27, 34, 35, 39].

Table 1. Windows event types and Unix event severities.

	Windows Event Type	**Unix Event Severity**	**Action Required?**
Error Threshold	Error	0 - Emergency 1 - Alert 2 - Critical 3 - Error	Yes, immediately
Warning Threshold	Failure Audit Warning	4 - Warning 5 - Notice	Yes, in near future
Informational Threshold	Success Audit Information	6 - Informational 7 - Debug	No

To compare event types and severities of both platforms we categorized them into three threshold levels.

Error Threshold. This threshold represents an application, service or system that is unavailable or unstable. An action is required to restore affected component(s). Sample events by platform are as follows:

- On Windows, error event is logged when an application cannot establish connection with remote resource or system could not start the service.
- In Unix, there are four levels of this threshold. Emergency severity (keyword panic) event is used when system becomes unusable and responsible support teams need to be engaged immediately. Events with alert severity (keyword alert) are logged for loss of primary connection link and restoration action must be taken immediately. Events with critical severity (keyword crit) also require immediate action. However, they directly do not impact systems' functionality, for example loss of redundant network link. Error severity (keyword err) event is used for non-urgent failures, where restoration action should be taken in given period.

Warning Threshold. In security events, this threshold is used to monitor failures on audited processes, for instance incorrect password provided in user login attempt. In system events this threshold is aligned to the alert configuration setting, for example disk utilization is above 80%. Sample events by platform are as follows:

- On Windows, failure audit events are registered when user is unable to access network folder. Warning type event is used in situations like TCP/IP has reached the security limit imposed on the number of concurrent TCP connect attempts.
- In Unix, warning severity (keyword warn) events indicate condition that may change to error threshold if action is not taken, for example file system utilization is currently at 80% where error severity for utilization is set 90% and more. Notice severity (keyword notice) event is used to log unusual application or system behavior that is not in error threshold. These events may suggest potential problems in the near future.

Informational Threshold. In security events it illustrates successful access attempt to audited processes, for example login to the system. In system events it shows successful operation of an application or service. It can also inform about change in alert configuration setting, for example CPU utilization declined below warning threshold level. This threshold does not require any action by support teams. Sample events by platform are as follows:

- On Windows, success audit event is logged for successful login to an application or system. In system events information event is used for successful driver load or periodic operating system uptime.
- There are two levels of this threshold in Unix. Informational severity (keyword info) event for logging normal operations like successful package installation. Debug severity (keyword debug) event is used to log additional details like application stack traces. Those details are very helpful when debugging the application behavior.

2.3. Polling Intervals and Data Retention

As part of design process of monitoring solution, organization needs to define monitoring polling intervals and data retention. Based on those, a policy should be established and

shared with the department that deploys and maintains monitoring system activities. Requirements and recommendations should follow industry standards and actual criticality of the systems available in the organization. The polling intervals have direct impact on monitoring system performance, the amount of data that is collected and the mean time to detect (MTTD) a failure in distribute systems infrastructure.

Table 2. Example of polling intervals and monitoring data retention.

	Standard System	Major System	Critical System
Availability monitoring	Every 10 minutes	Every 5 minutes	Every 1 minute
Capacity monitoring, for example CPU utilization	Every 20 minutes	Every 10 minutes	Every 5 minutes
Security and system logs [7]	Every 3 to 24 hours	Every 15 to 60 minutes	Every 5 minutes
Monitoring data retention [7]	1 to 2 weeks	1 to 3 months	3 to 12 months

Table 2 shows an example of polling intervals and data retention based on systems criticality. By *Standard System* we assume a system that has none or low impact on the organization, for example test server. *Major System* is defined as a system that would have moderate impact on the organization's productivity, for example through unavailability of communication channels such as internal news portal, email system or VOIP telephony system. *Critical System* is classified as it would have highest impact, for example on revenue in e-commerce organizations if front-end web servers become unavailable, or on organization's reputation if the system allowed an unauthorized access.

3. Areas of Monitoring

Within distributed systems infrastructure there are multiple metrics that can be collected. The key area monitoring solution was designed to monitor is systems availability and capacity.

3.1. Availability Monitoring

Availability monitoring provides details about systems, services and application accessibility. It is calculated as a percentage of time when system was running and accessible to the duration for which this metric is measured. Availability can be assessed by

analyzing results of ping command, verifying if process is in running state, dedicated port TCP/UDP is open or if application accepts user credentials during authentication process.

Availability is one of key indicators in Service Level Agreement (SLA) when organization provides IT functions as services. Sample visualization of LDAP service availability is shown in Figure 2.

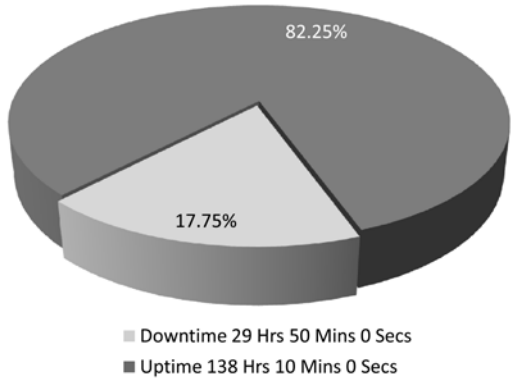

Figure 2. ManageEnginge AppManager - sample availability chart for LDAP service monitoring. Availability was measured for 7 days.

Today, multiple systems are designed to operate on a 24 x 7 basis. However it is very difficult to achieve 100% availability on a yearly report. Systems can be running although users are unable to access them due to for example network outages, required critical operating system updates or application code releases. Table 3 indicates the time of unavailability (also known as downtime) and its impact on a monthly and yearly availability report [21].

Table 3. Impact of downtime duration on availability metric.

Downtime per month	Downtime per year	Availability %
72 hours	36.5 days	90% ("one nine")
7.20 hours	3.65 days	99% ("two nines")
43.8 minutes	8.76 hours	99.9% ("three nines")
4.38 minutes	52.56 minutes	99.99% ("four nines")
25.9 seconds	5.26 minutes	99.999% ("five nines")

3.2. Capacity and Performance Monitoring

Second popular metric that monitoring solution provides is capacity and performance. As distributed systems are growing and continue utilizing more computer resources (see Figure 3), this metric helps forecasting future demands on computing power. Traditionally this

metric includes CPU utilization, memory utilization, storage (space used and bandwidth performance), network (bandwidth utilization) as well overall number of devices in the environment.

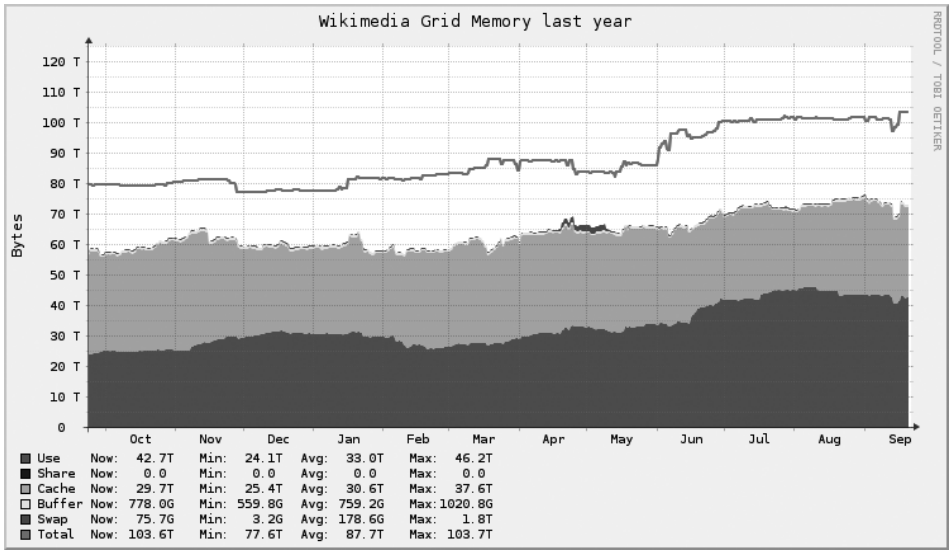

Figure 3. Ganglia - sample capacity chart based on Wikipedia grid memory utilization [28], September 2015 till September 2016.

Based on gathered capacity and performance metrics, IT departments can support business decisions when allocating budget on IT equipment. Moreover, collected metrics help application and service owners identifying systems that are overutilized (indication of further investment or architecture re-design) or systems that are underutilized (indication of application re-allocation, sharing of resources or application decommission).

3.3. Security Events Monitoring

Monitoring solution also covers security events monitoring. The market offers dedicated solutions within security information and event management (SIEM), however all infrastructure and application monitoring tools have a customized module, which can provide basic security events verification [8]. The main purpose of monitoring solution is to collect events from security and firewall logs, store and analyze them. Sample events include unauthorized access attempt due to a wrong password or missing permissions and distributed denial of service (DDoS). Monitoring solution can also detect increased number of login attempts from single location and acknowledge firewall rules are blocking miscellaneous network traffic.

4. Monitoring Approaches

There are two popular monitoring approaches, agent-based and agentless. Recently new methods are being introduced that encompasses agent-based and agentless advantages into one hybrid approach [8] or collection of monitoring metrics through data streams.

4.1. Agent-based Approach

An agent-based approach is platform dependent and requires additional software on monitored systems. It provides in-depth monitoring data as agents are domain specific and are designed to collect every possible metric. On the other side, this introduces limitation in scalability as the solution cannot be easily deployed in organization that uses multiple platforms, systems and applications. Maintenance and technical support of agent software can also be difficult as for example code upgrade needs to be performed on every server where the agent was installed. Overall architecture of agent-based approach in typical application, database and web server environment is presented in Figure 4.

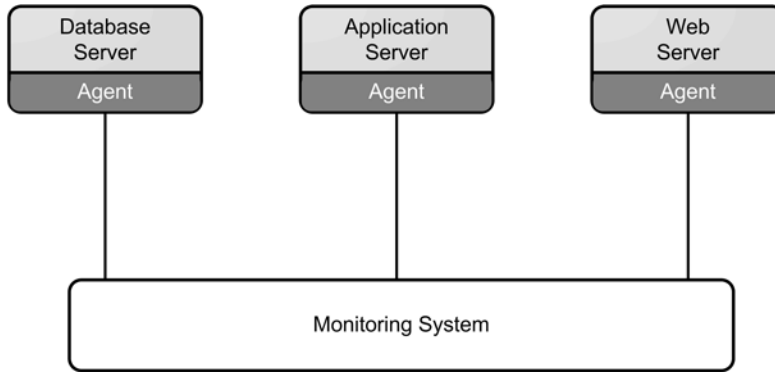

Figure 4. Agent-based approach architecture. Dedicated agent is installed on monitored system.

4.2. Agentless Approach

An agentless approach utilizes systems built-in monitoring technologies and protocols such as Windows Management Instrumentation (WMI) and widely available Simple Network Management Protocol (SNMP). It is a lightweight solution as it doesn't require additional software to be installed as well it is much easier to deploy in distributed environment. As shown in Figure 5 agentless approach provides systems availability monitoring without additional modules like it is required in agent-based approach.

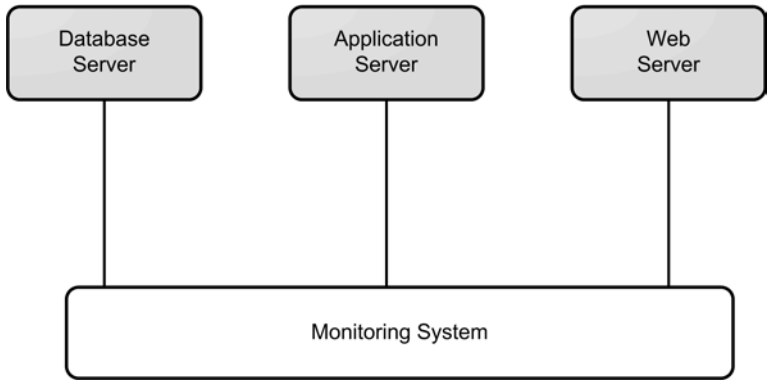

Figure 5. Agentless approach architecture. Monitoring system uses systems built-in monitoring protocols and technologies. No additional software is required.

However, an agentless approach is limited to generic monitoring metrics. When more granular monitoring data are required an additional diagnostic tools would need to be used.

4.3. Hybrid Approach

A hybrid approach provides new way of collecting data as it combines benefits of both agent-based and agentless approaches [8]. To meet all monitoring requirements it allows choosing between traditional monitoring approaches as well as gives an interface to integrate with custom monitoring scripts and agents (see Figure 6). This enables full flexibility and scalability to the current and future size and variety of monitored systems in a distributed environment.

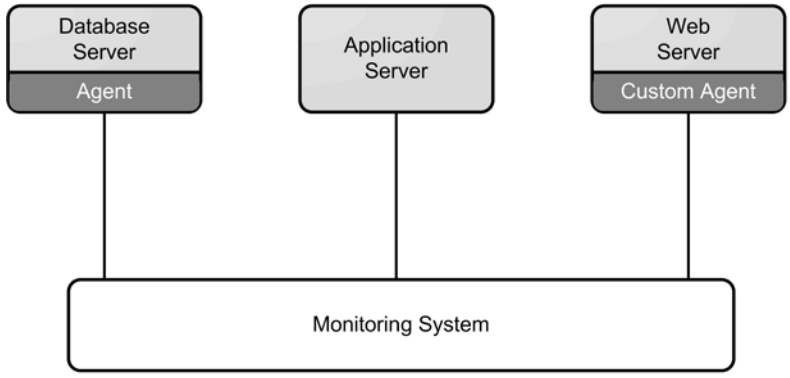

Figure 6. Hybrid approach architecture. Each monitored system has the best suitable monitoring approach deployed.

In hybrid approach, agent-based monitoring can be installed on mission critical systems where frequent and in-depth monitoring data are required to minimize applications downtime. A lightweight agentless approach can be employed on standard systems where only basic metrics like systems availability, CPU and disk utilization are needed. As hybrid

approach uses relevant traditional technique it allows to better adjust to the business needs and to sustain with an infrastructure growth.

4.4. Data Streams Approach

With evolution of distributed systems to heterogeneous and cloud environments, a new approach was developed to measure availability and performance from application and service perspective. This approach focuses on business transaction execution and validates end-to-end path from user initiating the task to the connected systems in the infrastructure. In order to accomplish this goal, software developers need to integrate data forwarders code into applications' code.

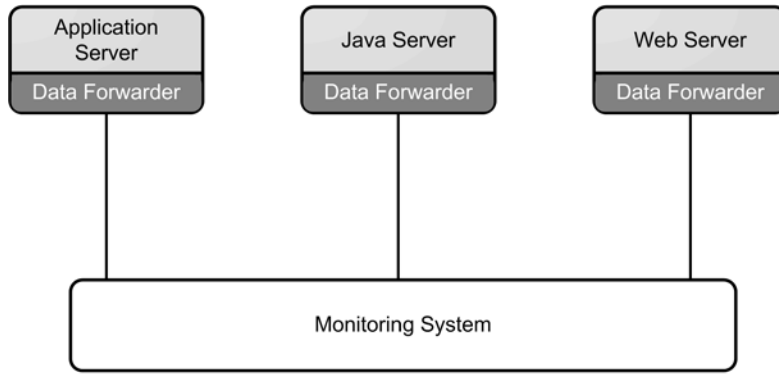

Figure 7. Data streams approach architecture. The data forwarder acts as an agent that transmits monitoring metrics as a stream.

The data forwarder is installed as an agent on application server, Java server or web server (see Figure 7). It transmits monitoring metrics as a stream to monitoring system. This allows IT operation teams watch application performance in near real time dashboards, create alerts based on trend lines and understand user's performance experience with the application or service. The data streams approach is similar to agent-based approach with the exception it was designed to be more dynamic and present overall systems performance rather than focusing on individual infrastructure components.

4.5. Overview of Monitoring Approaches

We examined eight characteristics of traditional monitoring approaches in comparison with new hybrid and data streams approach. This includes:

- *Platform dependency.* This characteristic is mainly dependent on additional software that needs to be installed on monitored system. The major platforms are Windows and Unix, as well as network and SAN / storage environments. When approach is platform dependent it limits its scalability and support of heterogeneous systems.

- *Availability monitoring.* This feature allows the monitoring approach to report on systems accessibility and operational status. It does not only include general check such as response to *ping* command; many systems require dedicated port to be open or system processes and services to be in running state.
- *Capacity monitoring.* Key elements for this characteristic include CPU utilization, available disk space and memory resources. In today's fast growing digital era it is very important to estimate and maintain basic capacity plan in order to accommodate future data demands. Based on chosen monitoring approach, this can be accomplished on general or in-depth level. Additionally, systematic data collection will allow identifying under- and overutilized systems and applications.
- *Alerting and notifications.* Apart from data collection the monitoring approach needs to detect anomalies and inform relevant support teams. The alerting mechanism includes setting state conditions / thresholds to report if system is up and running or failure has been detected. Notification methods are usually defined as sending an email, execution of SNMP trap or execution of custom script and remote command.
- *Monitored data granularity.* When monitoring system is being designed and deployed to monitor mission critical applications it is recommended to gather as much monitoring data as possible. This would allow quicker root cause analysis when issue occurs. The ultimate size and retention of gathered data needs to be discussed and took into consideration.
- *Monitored data gathering mode.* The monitored data are transferred from monitored systems to monitoring solution through Push, Request and Response and Pull modes. The Push mode is used based on time- or event-driven situations. In this mode data are sent (usually by an agent) from monitored system to central monitoring solution. In Request and Response mode the data are only transferred when monitoring solution requests for them. This mode is typical in agentless approach that uses built-in monitoring protocols and technologies. Pull mode is invoked by the monitoring solution and it collects already prepared data or data sets from the monitored systems. The data and data sets are arranged by a local agent or a script running as a scheduled job.
- *Additional software required on monitored systems.* When monitoring approach requires additional software on monitored systems the initial deployment of entire monitoring solution will require much more time comparing to approaches that use built-in monitoring protocols and technologies. The additional software is platform and monitored system dependent. For example Oracle database monitoring agent does not support Microsoft SQL system, similar to Windows operating system agents cannot be installed on Unix based platform. In certain scenarios, the approach may require dedicated appliance in order to monitor network devices.
- *Solution type.* This feature shows impact on capacity usage of monitored resources, such as processor, disk and memory utilization. It is recommended to adjust the monitoring approach to the current system performance that it does not have any negative impact.

- *Deployment and maintenance*. This characteristic presents the level of difficulty when deploying and maintaining the monitoring approach. This takes into consideration day-to-day operations and support, future software updates, configuration and simultaneous deployment across large environments, for example with 500 systems and more.

Results of our review are presented in Table 4.

Table 4. Overview of monitoring approaches.

	Agent-based approach	**Agentless approach**	**Hybrid approach**	**Data streams approach**
Platform dependency	Yes	No	Yes	Yes
Availability monitoring	No	Yes	Yes	Yes
Capacity monitoring	Yes	Yes	Yes	No
Alerting and notifications	Yes	Yes	Yes	Yes
Monitored data granularity	In-depth, full	General, limited	In-depth, full	In-depth, full
Monitored data gathering mode	Push	Request and Response	Push, Request and Response, Pull	Push
Additional software required on monitored systems	Yes	No	Yes - for in-depth data No - for standard data	Yes
Solution type	Heavy, lightweight	Lightweight	Lightweight	Lightweight
Deployment and maintenance	Difficult	Easy	Intermediate	Intermediate

5. How to Select the Tool?

Choosing the right tool to monitor distributed systems depends on four major factors. The most important is the *price* of the solution. It is not only the software license cost, but the decision maker should also take into account vendor consultation fees, staff training, documentation updates and time required to deploy it across entire domain. If the infrastructure is large, for example 1,000 servers, it may also be needed to hire dedicated personnel to maintain the monitoring solution.

The second factor is *functionality and scalability*. The market has many tools to offer and some of them give comprehensive solutions for heterogeneous environments while others are domain and platform specific. As many organizations started to use distributed

systems in the cloud environment this aspect needs also to be considered. Popular mechanism of up and down scaling infrastructure resources in the cloud introduces new challenges to traditional monitoring approaches. For example, agentless approach monitoring web server in the cloud may report the system as unavailable when the instance is automatically destroyed. False alerting can be prevented however it would require manual intervention in monitoring system. For systems with short lifespan (less than hour) this will be overwhelming people resources and mistakes are possible. In order to monitor dynamic resources, it is suggested to use data streams approach which focuses on business transactions performance rather than on individual system resources. It is also recommended to perform a research for which systems, services and applications will the selected monitoring solution be implemented as well which metrics are important to business decision maker. The number of selected metrics to be monitored has a direct impact on how much data are being gathered and stored for further analysis. The metrics may include infrastructure capacity planning, SLAs on systems availability etc. [6][11]. When monitoring solution is installed in one geographic location and monitored systems are in the other location(s), it is recommended to perform at least a week long network packets analysis on monitoring sample systems. This would give a good indication how wide area network (WAN) bandwidth utilization will increase when monitoring solution is fully deployed.

Next key factor is *alerting and integration* with existing systems. Usually when organization decides to deploy monitoring solution there are already many IT management tools such as service desks, support team email mailboxes, SMS notifications and user portals. When chosen solution offers customization the integration process will be seamless and easier to support.

Final factor is *deployment and maintenance*. There are tools available on the market which can be easily downloaded from vendor's website and installed in minutes. In some cases on-site vendor consultation and training will be required as solution may need dedicated software agents and configuration profiles. Large infrastructure environments will benefit when the monitoring solution can automatically discover configuration of systems to be monitored and install required agents when relevant. Also, it is important to discuss day-to-day maintenance of the monitoring solution as well as future vendor support with software upgrades when new version of the tool is released.

6. Monitoring Tools for Distributed Systems

There are multiple commercial and non-commercial tools available on the market. They monitor distributed systems using agent-based, agentless approach, hybrid approach and today more frequently with data streams approach. Here, we examine 15 tools (see Table 5).

Table 5. Monitoring tools for distributed systems.

Tool	License	Monitoring approach	Alerting	Cloud / Custom.	Target market size	Unique feature(s)
Zabbix	Open source, proprietary	Agent-based and agentless	Email, SMS, custom	Yes / Yes	Large and enterprise	Auto discovery, multiple plugins
Nagios	Open source, proprietary	Agent-based and agentless	Email, SMS, custom	Yes / Yes	Small, medium and large	Multiple plugins, wide community support, community customized versions
Ganglia	Open source	Agent-based	Optional via Nagios	Yes / Yes	Large and enterprise	Clusters and grid support
Hyperic	Open source, proprietary	Agent-based and agentless	Email, SMS	Yes / Yes	Small and medium	Easy deployment and configuration
ManageEngine AppManager	Proprietary	Agentless	Email, SMS, custom	Yes / Yes	Small and medium	Quick and easy deployment, application performance monitoring
IBM SmartCloud Monitoring	Proprietary	Agent-based and agentless	Email, SMS	Yes / Yes	Enterprise	Predictive analysis and reports
HP Operations Manager	Proprietary	Agent-based and agentless	Email, SMS, custom	No / Yes	Enterprise	Integration with other HP products
AppDynamics	Proprietary	Data streams	Email, SMS, API	Yes / Yes	Large and enterprise	Software as a Service
Datadog	Proprietary	Data streams	Email, SMS, API	Yes / Yes	Small, medium and large	Supports multiple cloud platforms and DevOps teams collaboration
Graphite	Open source	Data streams	No	Yes / Yes	Large and enterprise	Can handle 160,000 distinct metrics per minute

New Relic	Proprietary	Data streams	Email, SMS, API	Yes / Yes	Small, medium and large	Software as a Service, synthetic monitoring
Prometheus	Open source	Data streams	Email, SMS, API	Yes / Yes	Small, medium and large	Active community of developers and users
Riemann	Open source	Data streams	Email, SMS, API	Yes / Yes	Small and medium	Wide community support
Sensu	Open source, proprietary	Agent-based	Email, SMS, API	Yes / Yes	Small, medium and large	Configuration files automation via Chef and Puppet
TICK by InfluxData	Open source, proprietary	Data streams	Email, SMS, API	Yes / Yes	Small, medium and large	Software as a Service

Tools with open source *License* would usually have wide community support and possibilities to extend to enterprise model like Nagios. Proprietary tools would typically require vendor consultation when deployment is planned on large environment, for example more than 500 systems. *Monitoring approach* characterizes how the monitoring data will be collected. When monitoring system detects failure *Alerting* mechanism will be used to notify relevant support teams. Various alerting methods allow better alignment to organization needs and easier integrations with existing support engagement tools. Many tools have now *Cloud support* feature which allows further monitoring integrations with external providers. All the tools we examined had some *Customization* options such as custom script execution, formatting of email alert or opportunity for custom plugins development. *Target market* specifies the size of the environment the tool can support and its *Unique feature(s)* were provided in the last column.

6.1. Zabbix

Zabbix is open source software with great set of features that can be used in large and enterprise environments [40]. The application monitors servers, network devices, applications, databases and VMware virtual machines using agent-based and agentless approaches. Zabbix agent runs as native system process and does not require any specific environment like Java or .NET. Furthermore, Zabbix provides hardware monitoring for systems with Intelligent Platform Management Interface (IPMI) which gathers details about temperature, fan speed, chip voltage, and disk state.

Installation process is relatively easy. However, configuration and maintenance can be complex. Even though Zabbix is an open source application, it offers commercial support with custom features development, official training and professional consultancy.

6.2. Nagios

Nagios first launched in 1999 and now is one of the best-known open source system for monitoring IT infrastructure, systems, applications, services and business processes [30]. It is available in two versions, free Nagios Core, and commercial Nagios XI. Nagios Core has limited set of features for monitoring critical IT components; it can send alert notification by email, SMS or run custom script. Its web interface is very simple and provides reporting capabilities such as record of historical outages, events and alert notifications.

Nagios XI has all the monitoring features and easy-to-navigate web interface. This includes interactive dashboards with hosts overviews, services and network devices. Nagios XI provides improved reporting module of performance and capacity planning which helps organizations plan future infrastructure upgrades. Monitoring configuration of particular system can also be easily audited as the application offers configuration snapshot module that regularly saves the current configuration.

Nagios has wide community support with active development of new plugins and help with product installation, configuration and maintenance. The plugins especially allow Nagios Core users to expand monitoring capabilities as well as to adapt to new technologies, applications and systems without major software update.

Moreover, based on Nagios open source code some developers created their own monitoring solutions. Those solutions include Shinken and Icinga, and continue to be open source projects [25, 38].

6.3. Ganglia

Open source Ganglia was designed at Berkeley academic campus [19]. It primary objective was to collect metrics in near real time for large distributed systems such as grids and cluster up to 20,000 hosts where CPU utilization needs to be monitored every 10 seconds [9]. Ganglia provides in-depth operating systems metrics and using sFlow agent it can also gather information from network gear such as routers and switches. It uses XML language for data representation, External Data Representation (XDR) standard [18] for compact data transport and RRDtool for data storage and visualization. Implementation in Unix/Linux environment is robust. However, on Windows systems it requires (and is limited by) Cygwin libraries.

It is worth mentioning that Ganglia is not only aimed for university campuses. It has been successfully implemented by multiple companies in public and private sectors.

6.4. Hyperic

Available in two versions, open source Hyperic HQ and paid version vFabric Hyperic [23]. It was designed to monitor virtual and physical environments through agent-based and agentless approaches. Its key component, Hyperic Agent, automatically discovers system metrics such as memory, CPU utilization as well as applications being hosted on that system. Discovered resources are presented in Hyperic User Interface (aka Portal). Apart

from metrics collection Hyperic provides interface to remote resource control. User can start and stop application service or perform housekeeping functions on the database.

Hyperic installation and configuration is relatively easy and takes a few minutes. Alert notifications can be delivered as email, SNMP trap, SMS or integrated with other incident management system.

The vFabric Hyperic is a commercial version that offers enterprise support and has more features for example automated corrective actions.

6.5. ManageEngine AppManager

Installation and configuration of ManageEngine AppManager is very easy and intuitive [29]. The product is available in three price categories, free of charge when monitoring five servers and two paid versions, depending on size of the infrastructure it needs to monitor as well as available features. The most advanced version (Enterprise) is highly scalable with failover capabilities and supports distributed systems monitoring in multiple geographic locations.

The company is continuously developing the product to provide monitoring for latest technologies and systems such as Cassandra and Couchbase databases, Docker solution for software distribution in virtual infrastructure or Hadoop clusters monitoring.

As AppManager supports multiple platforms it can easily be deployed in organizations with heterogeneous environment where hardware and operating systems come from different vendors. Another good feature is interface for execution of custom scripts that can collect any monitoring data.

6.6. IBM SmartCloud Monitoring

IBM SmarCloud Monitoring comes as a bundle of well established IBM Tivoli infrastructure management products which includes Tivoli Monitoring and Tivoli Monitoring for Virtual Environments [24]. It is easy to install although configuration and management requires some expert knowledge.

The new solution includes improved analytic modules, and capacity and reporting tools. It also provides dynamic usage trending and health alerts for all monitored resources – physical, virtual and in cloud.

User Interface is web based and offers user role-based views, such as cloud admin, capacity planner, and application's owner. Multiple built-in dashboards provide quick view to entire monitored infrastructure in addition to in-context menu which gives a user an opportunity to quickly drill down to relevant management tool.

6.7. HP Operations Manager

HP Operations Managers offers comprehensive solution for physical and virtual infrastructures [22]. It collects monitoring data using agent-based and agentless approaches as well as provides easy integration with third-party tools such as network monitors. The

built-in discovery module automatically recognizes managed nodes and configures monitoring rules which essentially allows quicker deployment of entire solution.

The tool operates as client-server solution and provides intuitive User Interface. Large deployments would require IT expertise. In order to collect detailed information about infrastructure, operating systems and applications HP Operations Manager requires additional plugins, so called Smart Plug-Ins (SPIs). This product was the only one which didn't support monitoring of public clouds, such as Amazon Web Services (AWS).

6.8. AppDynamics

AppDynamics offers a solution for applications performance monitoring that can be implemented in the organization's environment, delivered through Software as a Service (SaaS) model or in hybrid deployment [15]. It provides monitoring metrics using data streams approach. Installation is easy and requires minimal configuration. Alerting module in this solution uses machine learning algorithms to create dynamic baselines and to detect anomalies in the infrastructure technology stacks.

Apart from applications performance monitoring, AppDynamics can monitor web applications response time from real end-user's browser and mobile phone perspective. This can help optimizing page load times and detect errors through content checks on a web site.

6.9. Datadog

Datadog solution was primarily designed to monitor cloud environments and seamlessly integrate with collaboration applications used today by DevOps support teams [16, 17]. It works with more than 100 applications and systems that generate monitoring metrics. The agent that sends the data streams is supplied on open source license which gives additional flexibility to the organization.

Datadog gathers application performance data, presents metrics in intuitive dashboards and notifies support teams through various channels when failure is detected. It also aims for live collaboration with interactive dashboards and tools such as Hipchat and ChatWork for group discussions.

6.10. Graphite

Graphite is highly scalable, real-time charts rendering application, written to visualize monitoring metrics gathered by other applications [20]. Its main purpose is to store the numeric time-series data and render graphs of that data on demand. The solution is enterprise ready, can handle approximately 160,000 distinct metrics per minute. Graphite requires implementing additional plugins to provide alerting mechanism.

As an open source product, this solution can integrate with multiple data collection tools, data forwarders, data storage alternatives as well as other open source and paid monitoring solutions.

6.11. New Relic

New Relic monitoring solution is only available in Software as a Service (SaaS) model [31]. This gives many benefits from infrastructure maintenance and support perspective, however many organizations may raise a concern about security aspects and performance from remote locations, especially in Europe. Deployment of this solution is very easy and the system also supports on-premise applications.

The solution can monitor synthetic transitions from multiple geographic locations, provide guidance and code diagnostics on web applications and services. It has capabilities of analyzing application performance on mobile phone devices.

6.12. Prometheus

Prometheus is an open source project that collects monitoring metrics through data streams [33]. It stores all the data as time-series identified by metric name and key-value pairs. Each server node hosting Prometheus solution runs autonomously. All the collected data are available through dashboards that provide multiple visualization layers and integration with other applications. Data gathering process runs via pull mode over HTTP. Push mode is also supported through an intermediary gateway server.

Prometheus has active community of users and developers. Moreover, the community published best monitoring practices such as metric and label naming, creating dashboards, and configuring alerts on Prometheus website.

6.13. Riemann

Rieman solution is similar to Prometheus and stores data as time-series with powerful processing language [36]. It was designed for DevOps teams that support dynamic production infrastructure. Metrics are visualized using Graphite application while alerts and notifications can be integrated with PagerDuty platform [32].

With Riemann monitoring metrics can be visible within milliseconds comparing to traditional solutions that pull data every five minutes or less frequent.

6.14. Sensu

Sensu offers a solution for public, private and hybrid cloud computing to monitor servers, services, applications, and business KPIs [37]. It is available in two versions, open source Sensu Core and paid Sensu Enterprise. The latter version gives additional functionally, for example alert assignment to dedicated support team, built-in third party integrations, audit logging, enterprise-level dashboard, vendor support and annual trainings.

Sensu provides extensible framework including a message bus, event processor, monitoring agent, and documented API that allows quicker deployment in large and heterogeneous environments.

6.15. TICK by InfluxData

InfluxData created a monitoring platform that consists of Telegrapf (T), InfluxDB (I), Chronograf (C) and Kapacitor (K) [26]. All those components make up a TICK stack. Telegrapf is collecting data as time-series, InfluxDB delivers high performance database for writing time-series, Chronograf visualize the data and Kapacitor is responsible for alerting, ETL processes and anomalies in gathered time-series.

The solution is available as open source version and paid version through InfluxCloud. The latter is also available on AWS platform. The company offers professional services including developers support and training courses.

7. Time to Notify Experiment

Moreover to the review of currently available monitoring tools we conducted a monitoring experiment with three popular tools: Nagios XI, Prometheus and Sensu. The main goal of the experiment was to measure the time since failure occurred to notify support team about it. As a sample failure we chose situation when CPU utilization exceeds 95% threshold. The tools were selected by following criteria: a) support of cloud environments, b) wide community support, c) at least one tool with agent-based monitoring approach and one tool with data streams monitoring approach, and d) availability of open source version of the solution.

7.1. Tools Installation and Configuration

In the datacenter, tools were installed on individual servers running CentOS 6.8 operating system. Nagios XI provides the most seamless installation as it only requires downloading one file and executing one command. Once the command finishes, the user can login to Nagios XI web interface and start adding systems to be monitored. Alerting configuration is also simple and for example email notifications require only local or remote SMTP server and account details. To monitor local and remote servers, dedicated agent named Nagios Remote Plugin Executor (NRPE) needs to be deployed. The agent works as a communication gateway that allows remote scripts execution and sends results back to the monitoring solution. Monitoring of system's availability Nagios XI portrays as a 'host' while any metrics collected within that host is shown as a 'service'. Installations of Prometheus and Sensu were more advanced as those tools are built on modular components and the majority of their configurations is only available through text files.

Prometheus as complete monitoring solution requires installation of core module, Node Exporter module that gathers basic metrics of local server, PromDash module to build dashboards helping visualize metrics from multiple data sources and Alertmanager module to build various alert notifications. Each of those modules needs separate configuration file written in YAML language and uses dedicated TCP port for communication. YAML language makes the configuration files' format human-readable however is very sensitive and for example one extra white space in the line can make this line to be ignored or change

the logic how the alerts' rules are being processed. To avoid configuration mistakes in alerting module, Prometheus gives visual editor that shows alert's routing tree. As this tool has wide community support, many step-by-step tutorials are available including Docker images with complete solution. In order to monitor basic server resources, Node Exporter module needs to be installed on every local and remote server. Monitoring of dedicated resources like MySQL database requires additional module to be installed on the database server.

Sensu similarly to Prometheus has modular architecture and as a solution requires core module which includes client module, Redis data-store used as a database and cache, Rabbit MQ used a transportation and communication layer, and dashboard module. Redis and Rabbit MQ need to be installed before deploying Sensu package. Every module has its configuration file in JSON format and uses dedicated TCP port for communication. In order to monitor local and remote servers, Sensu package needs to be installed on every system. Sensu similarly to Nagios XI collects the data metrics by executing scripts; however scripts are initiated by local process rather than by the central process running on monitoring solution server like in Nagios XI. Once the script finishes, its results are sent to Sensu central server using communication bus (Rabbit MQ). The same bus is used for receiving instructions what needs to be monitored and how often. The monitoring and polling configurations are always kept on Sensu central server.

7.2. Experiment and Results

The experiment to measure time to notify support team since server's CPU utilization exceeds 95% threshold was conducted using systems in Amazon AWS cloud and Google Cloud. Servers in the clouds were running Linux or CentOS operating system including Nagios NRPE package, Prometheus Node Exporter, and Senus package with only Sensu Client service activated. Both of the cloud providers allowed configuring their network to be accessed only from a dedicated IP addresses such as the addresses of our monitoring solutions' servers. In order to generate high CPU utilization event we installed relevant cpuburn package that can constantly cycle through floating-point unit (FPU) intensive functions.

In the cloud environments cpuburn was scheduled to be executed every 12 or 15 minutes and start at random second of the minute. Each time the scheduled task ran, it ran for four minutes. Nagios XI and Sensu were configured to monitor servers' CPU utilization every one minute. Due to the design how Prometheus stores data in the time series and calculates monitoring metrics, it was set to monitor servers' CPU utilization every 30 seconds. All the tools were configured to generate email alert only when high CPU event lasted for at least one minute.

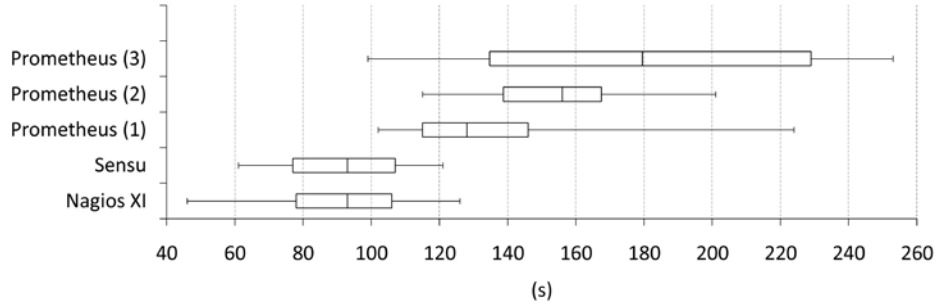

Figure 8. Time to notify support team since high CPU event occurred. There are three sets of Prometheus results due to various events grouping configurations (1) group wait 10s, group interval 2m, (2) group wait 30s, group interval 5m, (3) grouping disabled.

The results (Figure 8) gathered from more than 400 scenarios show that Nagios XI and Sensu median time to notify support team was 93 seconds since the CPU utilization exceeded 95% threshold. Prometheus Alertmanager module provides grouping mechanism that can prevent flooding of alert notifications when multiple resources report the same failure within short period of time. Prometheus group_wait variable allows the extra time to wait before sending initial notification, while group_interval holds specified time after initial notification to send a batch of new alerts that came for the same group. Due to that mechanism and the design how Prometheus calculates CPU utilization metric, we noticed significant delay in sending email notification comparing to Nagios XI and Senus. Prometheus median time to notify was 128 seconds (group_wait 10s, group_interval 2m), 156 seconds (group_wait 30s, group_interval 5m), and 179.5 seconds when grouping was disabled.

During experiment period we also noticed that Nagios XI and Sensu were unable to notify on high CPU utilization on some of Amazon AWS Linux micro instances. This occurred as the cloud provider was protecting other customers' instances and limiting CPU resources of our instances [14]. Nagios XI detected the limited CPU condition as 'CPU steal' metric however we were unable to set an alert condition on that metric (Nagios XI supports alert configuration only on CPU user, system and iowait metrics). We did not observe these issues with Prometheus as alerts were based on CPU idle metric. We also did not experience CPU limitations while running cpuburn on Linux instances in Google Cloud.

8. Conclusions

We reviewed 15 monitoring tools which can visualize current state of distributes systems infrastructure. They also provide excellent capabilities of notifying relevant support teams when failure has been detected. Experiment conducted during our review presents how quickly three popular monitoring solutions can send email notification with sample alert.

We discussed basic concepts of monitoring, areas where monitoring provides benefits and key factors that should be considered when choosing a monitoring solution. The future distributed systems computing will use cloud environments and it is recommended for

monitoring solution to support that capability [1, 3, 10]. From maintenance and data collection perspective we presented four approaches to choose from when designing and deploying a monitoring solution.

References

[1] Aceto G., Botta A., De Donato W., Pescape A., Cloud monitoring: A survey, Computer Networks, vol. 57, pp. 2093-2115, 2013.
[2] Boccia V. et al., Infrastructure Monitoring for distributed Tier1: The ReCaS project use-case, International Conference on Intelligent Networking and Collaborative Systems, Salerno, Italy, 2014.
[3] Fatema K., Emeakaroha V. C., Healy P. D., Morrison J. P., Lynn T., A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives, Journal of Parallel and Distributed Computing, vol. 74, no. 10, pp. 2918-2933, 2014.
[4] Hakulinen T., Ninin P., Nunes R., Riesco-Hernandez T., Revisiting CERN Safety System Monitoring (SSM), Proceedings of International Conference on Accelerator & Large Experimental Physics Control Systems, San Francisco, California, USA, 2013.
[5] Hernantes J., Gallardo G., Serrano N., IT Infrastructure-Monitoring Tools, IEEE Software, vol. 32, no. 4, pp. 88-93, 2015.
[6] Horalek J., Sobeslav V., Proactive ICT Application Monitoring, Latest Trends in Information Technology, Wseas Press, pp. 49-54, 2012.
[7] Kent K., Souppaya M., Guide to Computer Security Log Management, US Nat'l Inst. Standards and Technology, Sept. 2006; http://csrc.nist.gov/publications/nistpubs/ 800-92SP800-92.pdf.
[8] Kufel L., Security Event Monitoring in a Distributed Systems Environment, IEEE Security & Privacy, vol. 11, no. 1, pp. 36-43, 2013.
[9] Massie M., Li B., Nicholes B., Vuksan V., Monitoring with Ganglia, Book published by O'Reilly Media, 2013.
[10] Smit M., Simmons B., Litoiu M., Distributed, application-level monitoring for heterogeneous clouds using stream processing, Future Generation Computer Systems, vol. 29, pp. 2103-2114, 2013.
[11] Spellmann A., Gimarc R., Capacity Planning: A Revolutionary Approach for Tomorrow's Digital Infrastructure, Computer Measurement Group Conference, La Jolla, California, USA, 2013.
[12] Terenziani P., Coping with Events in Temporal Relational Databases, IEEE Trans. Knowledge and Data Eng., vol. 25, no. 5, pp. 1181-1185, 2013.
[13] Tierney B., Crowley B., Gunter D., Holding M., Lee J., Thompson M., A Monitoring Sensor Management System for Grid Environments, Proceedings of The Ninth International Symposium On High-performance Distributed Computing, IEEE CS, pp. 97-104, 2000.
[14] Amazon AWS Micro instance limitations, https://aws.amazon.com/ec2/faqs, Jul 2016.
[15] AppDynamics, Application Performance Monitoring & Management, http://www.appdynamics.com, Apr 2016.
[16] Datadog, Cloud Monitoring as a Service, http://www.datadoghq.com, Apr 2016.

[17] DevOps support teams, http://theagileadmin.com/what-is-devops, Apr 2016.
[18] External Data Representation (XDR), Wikipedia page, https://en.wikipedia.org/wiki/External_Data_Representation, Feb 2016.
[19] Ganglia Monitoring System, http://ganglia.sourceforge.net, Feb 2016.
[20] Graphite, Graphs rendering application, http://graphite.readthedocs.org, Apr 2016.
[21] High availability, Wikipedia page, https://en.wikipedia.org/wiki/High_availability, Feb 2016.
[22] HP Operations Manager, http://hp.com/go/Ops, Feb 2016.
[23] Hyperic Application & System Monitoring, http://sourceforge.net/projects/hyperic-hq, Feb 2016.
[24] IBM SmartCloud Monitoring, http://ibm.com/software/tivoli/products/smartcloud-monitoring, Feb 2016.
[25] Icinga, Open Source Monitoring, http://www.icinga.org, Apr 2016.
[26] InfluxData, The platform for time-series data, https://influxdata.com, Apr 2016.
[27] International Telecommunication Union, X.733: Information technology - Open Systems Interconnection - Systems Management: Alarm reporting function, http://www.itu.int/rec/T-REC-X.733/en, Apr 2016.
[28] Live monitoring console of Wikimedia Grid, http://ganglia.wikimedia.org, Feb 2016.
[29] ManageEngine Applications Manager, http://appmanager.com, Feb 2016.
[30] Nagios - The Industry Standard In IT Infrastructure Monitoring, http://www.nagios.org, Feb 2016.
[31] New Relic, Application Performance Management & Monitoring, http://newrelic.com, Apr 2016.
[32] PagerDuty, The Incident Resolution Platform For IT Operations & DevOps Teams, http://www.pagerduty.com, Apr 2016.
[33] Prometheus, Monitoring system and time-series database, http://prometheus.io, Apr 2016.
[34] Request for Comments (RFC) 5424 - The Syslog Protocol, http://tools.ietf.org/html/rfc5424#section-6.2.1, Feb 2016.
[35] Request for Comments (RFC) 5674 - Alarms in Syslog, https://tools.ietf.org/html/rfc5674.html, Apr 2016.
[36] Riemann, A network monitoring system, http://riemann.io, Apr 2016.
[37] Sensu, Monitoring for today's infrastructure, https://sensuapp.org, Apr 2016.
[38] Shinken Monitoring, http://shinken-monitoring.org, Apr 2016.
[39] Windows Event Types, http://msdn.microsoft.com/en-us/library/windows/desktop/aa363662.aspx, Feb 2016.
[40] Zabbix - The Enterprise-Class Open Source Network Monitoring Solution, http://www.zabbix.com, Feb 2016.

Received 22.02.2016, Accepted 15.09.2016