Cloud Monitoring

Eduard-Florin Hogea
Johannes Kepler University
K12019919
West University of Timisoara
Linz, Austria
eduard.hogea00@e-uvt.ro

Alacik Burak
Johannes Kepler University
K01603765
Linz, Asutria
alacik.burak@live.at

ABSTRACT

Cloud monitoring represents a crucial part of every IT infrastructure. It is used by collecting metrics, events, and metadata to efficiently keep track of the performance, compliance, and security, managing the workflow of them. In this thesis, we provide an explanation of what is Cloud monitoring giving some background description and presenting the available tools and applications of it while also discussing the main motivation behind the writing of this paper. Topics discussed in this are Cloud Monitoring, Virtual Machines, Containers, Paywalls and Applications, and Existing Tools for them.

KEYWORDS

Cloud, Monitoring, Paywall, Containers, Virtual Machines, Comparison

1 INTRODUCTION

Cloud monitoring is used more and more nowadays by consumers and providers because of its importance in managing IT infrastructures. By collecting data from the infrastructures, Cloud monitoring services can use it to generate insights, charts, or alerts which can be either manually or automatically dealt with. To help to cover the problem that we are going to discuss in this paper, we will give a short, informal explanation of why Cloud monitoring is important. By keeping full track of what has happened in infrastructure before and what is going on at this moment, one can more easily make the right decision for peaking the workflow. Aspects that can be improved by that would be:

- Performance
- Capacity and Resource Management
- Security

The problem that we are going to discuss in this paper, is the undeserved lack of recognition that Cloud monitoring gets. While being very important and also pretty widespread it has received limited attention from the research community with few exceptions and While there are lots of big providers, the term Cloud monitoring is not so well known or understood in the normal consumers.

The motivation behind the writing of this paper is to address the importance of Cloud monitoring with concrete examples of its usage in day to day life via the listing of most of its applications and the existing tools available and trying to give it the attention that it deserves.

2 BACKGROUND

For a better understanding of the importance of Cloud monitoring, we are also providing a more in-depth look. A formal explanation of what Cloud monitoring means can be found here [12] where it explains that it is the task of paramount importance for both Cloud Service Providers and Cloud Service Consumers. On the one side, it is a key tool for controlling and managing hardware and software infrastructures; on the other side, it provides information and Key Performance Indicators) for both platforms and applications.

The continuous monitoring of the Cloud and of its Service Level Agreements, - for example, in terms of availability, delay, etc. - supplies both the Providers and the Consumers with information such as the workload generated by the latter or the performance and Quality of Service offered through the Cloud, also allowing to implement mechanisms to prevent or recover violations, for both the Provider and Consumers. Cloud Computing involves many activities for which monitoring is an essential task.

2.1 Cloud Monitoring capabilities:

In order for a cloud monitoring tool to be classified as efficient, it must have certain capabilities.Some properties are also considered essential for cloud systems and are therefore also seen important and adopted for the monitoring tools, as these are ultimately active in the cloud environment and have to adapt. In the following we list some important skills that can also be used when performing complex management activities in a cloud environment. This results in the following properties, which can also be found in the marked journal [18]:

- Portability: To achieve efficiency, the monitoring tools must be capable of being transferred to other platforms. This means that the tools can be used on many different clouds so that they can be widely used in the market.

- Scalability: About resource and application management in clouds, developers are currently striving for a certain degree of scalability. It is therefore important to monitor the supervised delivery processes to ensure adequate management. The reason is that the cloud deployment can be very extensive and can consist of a large number of nodes. To manage these resources, a monitoring tool must be scalable to provide the monitored information in a timely and flexible manner.

- Robustness: The cloud environment is designed in such a way that it can change continuously, for example by adding new resources. It is therefore a task of the monitoring tool to perceive these changes and to react accordingly. By adapting to the new status, it is then possible that the tool can continue to carry out its operations without any errors.

- Non- intrusiveness: Since the number of resources in a cloud can be very large, and all of these should also be monitored, this can hurt the performance of the entire system. The system mustn't come to a standstill. Therefore, the monitoring tool is expected to use as little capacity as possible on the part of the system to be monitored, so that the performance does not fall behind.

- Interoperability: The resources of a cloud environment, which form a resource as independent data centers, are kept very high by the amount. With interoperability in terms of monitoring, it is understood that a modern monitoring tool can ensure the exchange of information between heterogeneous clouds to be able to manage collaborative operations. Interoperability is a prerequisite for the creation of offers from several providers.

- Multi-tenancy: Clouds can offer a so-called multi-tenant environment in which they share common physical resources and application instances. The necessity of this requirement has been discussed several times, especially in the area Guarantee of service level agreements and monitoring of the virtual machine. To support multi-tenancy, the cloud monitoring tool should be able to maintain parallelism. This enables several customers to receive jointly monitored information. Isolation plays an important role here, as clients only have access rights to the information addressed to them.

- Customizability: The goal that the numerous cloud providers are striving for today is to find a way, or rather to allow the customer to adapt so that the customer can decide for himself. It must also be possible for the customer to select the metrics that are used to monitor their services. Thus, the cloud must be able to provide the service adaptation for all of its customers. This ability is expected of all efficient tools today.

- Extensibility: Since this area, i.e. cloud computing, is a very rapidly growing area, there are numerous changes and extensions to technologies, especially in the area of management. Monitoring is fundamental in the management of cloud computing. The tools are therefore expected to be able to adapt quickly to changes and new environments and to be able to make new metrics their own. This is achieved mostly through a modular design.

- Usability: The intention of the tool plays a major role here since it forms a standard for qualification and determines usability. In the area of cloud monitoring, usability is understood to mean the suitability of the tool that was designed to support cloud management. Funding in the area of human interaction, maintenance, and deployment, can contribute to high accessibility.

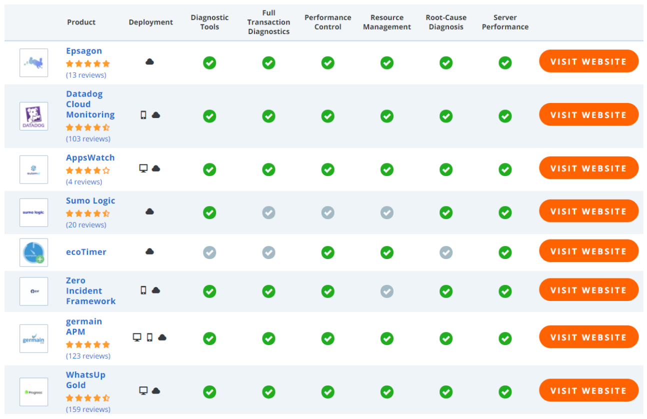
- Affordability: Keeping costs as low as possible is always welcomed in a company and operations. For this reason, customizing the cloud is a current and popular topic. The cheaper a tool is, the higher the likelihood of its widespread use and acceptance. This is why many monitoring tool users and companies use open source to reduce costs. So if you offer the tool as open-source, the likelihood of a wide expansion of the product is very high.

- Shared resource monitoring: The virtualization of physical resources is used in the field of cloud computing to both increases the performance of the services and to accomplish the isolation. The process of virtualization works in such a way that the virtual machines share the resources on which they are based and the applications use the resources of the virtual machines. Efficient supervising of these conflicts is required to avoid possible conflicts. The monitoring serves on the one hand to avoid conflicts that can arise among the resources of the virtual machines, and on the other hand to manage the applications that share these virtual resources. Therefore, cloud monitoring tools are expected to also monitor shared resources to prevent such conflicts.

- Archivability: The monitoring tool is expected to be able to archive historical data so that it can be used in the event of malfunctions. This can be very helpful in the analysis to find the core of the problem.

3 APPLICATIONS AND EXISTING TOOLS

Some of the applications [2][7][6][5][4][10][11] that we were able to find with the help of [8], followed by a short description provided are, as seen in the Figure 1:



Product	Deployment	Diagnostic Tools	Full Transaction Diagnostics	Performance Control	Resource Management	Root-Cause Diagnosis	Server Performance	
Epsagon ★★★★★ (13 reviews)	☁	✅	✅	✅	✅	✅	✅	VISIT WEBSITE
Datadog Cloud Monitoring ★★★★⯪ (103 reviews)	📱 ☁	✅	✅	✅	✅	✅	✅	VISIT WEBSITE
AppsWatch ★★★★☆ (4 reviews)	🖥 ☁	✅	✅	✅	✅	✅	✅	VISIT WEBSITE
Sumo Logic ★★★★⯪ (20 reviews)	☁	✅	✓	✓	✓	✅	✅	VISIT WEBSITE
ecoTimer	☁	✓	✓	✅	✅	✓	✅	VISIT WEBSITE
Zero Incident Framework	📱 ☁	✅	✅	✅	✓	✅	✅	VISIT WEBSITE
germain APM ★★★★★ (123 reviews)	🖥 📱 ☁	✅	✅	✅	✅	✅	✅	VISIT WEBSITE
WhatsUp Gold ★★★★⯪ (159 reviews)	🖥 ☁	✅	✅	✅	✅	✅	✅	VISIT WEBSITE

Figure 1: Comparison of applications

The applications provide more or less the same thing. While it may differ, the main features that these applications provide are:

- Product Deployment
- Diagnostic Tools
- Full Transaction Diagnostics
- Performance Control
- Resource Management
- Root-Cause Diagnosis
- Server Performance

3.1 The price of Cloud monitoring

To realize the basic idea of the cloud, which results from the fact that you only pay for what you have used, it is necessary to record and select the usage information. Only then is it also possible to adequately offset the services used by the user. This also includes, for example, the consumption of virtual resources and applications, of which the usage information plays a role. The metrics used for recording can be, e.g. the amount of bandwidth used, the computing hours used, etc.

It is therefore important to ensure that an accounting system that is as transparent as possible, is made available. The system makes it possible that data can be recorded in a trustworthy, secure, and verifiable manner. This is also primarily required to ensure protection against external influences that can lead to incorrectly modified changes to the data. To counteract these problems, a monitoring system, which exhibits robust and secure monitoring capabilities, is required. We can conclude, that one of the important usage areas of cloud monitoring, is to ensure safety. The user is given the ability to extract his usage information and can conclude from this whether the current usage is normal or excessive. This was done with the help of [18] information provided.

3.2 Cloud monitoring systems

Cloud monitoring systems are those systems that have been planned from the beginning for monitoring cloud deployments. Usually, those systems are aware of the cloud concepts discussed such as elasticity, availability, scalability, interoperability, customizability, and extensibility. Theoretically, these tools have advantages over the non-cloud-aware tools as they can utilize cloud properties to scale and adapt as the system they monitor changes. Most of the systems in this class are recent developments and lack the user base, support, and range of plugins that are available for many earlier systems, This presents a huge trade-off which will likely lessen as time passes.

Some examples are cloudinit([1]), Sensu([3]), Konig etal([19]), DARGOS([22]), CloudSense([20]), GMonE([21]), Lattice([16]) and Varanus([23]). We took overview of them from the products website and [24]

Cloud-init is the industry standard multi-distribution method for cross-platform cloud instance initialization. It is supported across all major public cloud providers, provisioning systems for private cloud infrastructure, and bare-metal installations.

Cloud instances are initialized from a disk image and instance data:

- Cloud metadata
- User data (optional)
- Vendor data (optional)

Cloud-init will identify the cloud it is running on during boot, read any provided metadata from the cloud and initialize the system accordingly. This may involve setting up the network and storage devices to configuring SSH access key and many other aspects of a system. Later on the cloud-init will also parse and process any optional user or vendor data that was passed to the instance.

Sensu is a complete solution for monitoring and observability at scale. Sensu Go is designed to give you visibility into everything you care about: traditional server closets, containers, applications, the cloud, and more. Sensu is an agent-based observability tool that you install on your organization's infrastructure. The Sensu backend gives you a flexible, automated pipeline to filter, transform, and process alerts and metrics.

Sensu Go is operator-focused and developer-friendly and integrates with popular monitoring and observability tools. Deploy Sensu Go for on-premises and public cloud infrastructures, containers, bare metal, or any other environment.

Konig et al. propose a distributed peer to peer tool for monitoring cloud infrastructure is a three level architecture consisting of a data, processing and distribution layer. The data layer is responsible for obtaining monitoring data from a range of sources including raw log files, databases and services in addition to other conventional monitoring tools including Ganglia and Nagios. The processing layer exposes a SQL-like query language to filter monitoring data and alter system configuration at run time. The distribution layer operates a peer to peer overlay based on SmartFrog [72] which distributes data and queries across the deployment.

Konig et al., 2012 is similar to other p2p monitoring systems in terms of topology but introduces a powerful query language for obtaining monitoring data as opposed to a conventional, simple REST API. Additionally the data layer component abstracts over numerous data sources, including other monitoring systems making this work a notable tool for building federated clouds and other architectures which encapsulate existing systems

DARGOS Distributed Architecture for Resource manaGement and mOnitoring in cloudS (DARGOS) is a fully decentralized resource monitor. DARGOS, like SQRT-C, makes use of the OMG Data Distribution Standard (DDS) to provide a QoS sensitive pubsub architecture. DARGOS has two entities: the Node Monitor Agent (NMA) and the Node Supervisor Agent (NSA) which for all intents and purposes are the publisher and subscriber, respectively. The NMA collects state from the local VM and publishes state using the DDS. The NSA is responsible for subscribing to pertinent monitoring state and making that state available to analysis software, users, or other agents via an API or other channel. DARGOS includes two mechanisms to reduce the volume of unneeded monitoring state which is propagated to consumers: time and volume-based filters. These mechanisms reduce unnecessary traffic and yield improvements over other pubsub schemes.

While DARGOS makes use of the DDS standard, QoS is not its primary concern and no specific provisions are made to ensure the low latency and jitter achieved by SQRT-C. Conceptually, DARGOS is similar to other pubsub based monitoring tools including GMOnE, SQRT-C, and Lattice

CloudSense is a data center monitoring tool that, unlike other tools surveyed here, operates at the switching level. CloudSense attempts to address networking limitations in previous tools which prevent the collection of large volumes of fine grain monitoring information. CloudSense operates on switches and makes use of compressive sensing, a recent signal processing technique that allows for distributed compression without coordination, to compress the stream of monitoring state in the network. This allows CloudSense to collect greater monitoring information without using additional bandwidth. In the proposed scheme each rack switch in a data center collects monitoring state from servers within each rack and compresses and transmits aggregated state to a master server which detects anomalous state.

CloudSense is primarily intended for datacenter monitoring and uses a compression scheme which lends itself well to anomaly detection. CloudSense has been proposed as a tool for monitoring MapReduce and other performance bound applications where anomaly detection is beneficial. This architecture in itself, is not enough to provide a comprehensive monitoring solution but does present a novel option for anomaly detection as part of a larger tool.

GMonE is a monitoring tool that covers the physical, infastructure, platform, and application layers of the cloud stack. GMonE attempts to overcome the limitations of existing monitoring tools which give only a partial view of a cloud system by giving a comprehensive view of all layers of the cloud. To this end, a monitoring agent: GMonEMon is present in all monitored components of the cloud stack including physical servers, IaaS VMs, PaaS instances, and SaaS apps. GMonEMon has a plugin architecture that allows users to develop additional modules to obtain metrics from all relevant sources. GMonEMon uses its plugins to collect data and then uses Java RMI based publish-subscribe middleware to publish monitoring state. A database component: GMonEDB acts as a subscriber to various GMonEMon instances and stores monitoring state using MySQL, RRDtool, Cassandra, or other back end.

Unlike the vast majority of monitoring tools surveyed here, GMondE provides monitoring services to both cloud providers and cloud users. In a GmonE deployment there are at least two GMonE DB instances: one for a user and one for the provider. Each stakeholders subscribe to the components relevant to their operations. This could be used to allow users to obtain monitoring state from the layers bellow the layer that they are operating on. For example, as SaaS could use this scheme to obtain metrics regarding the PaaS platform, the VMs running the platform, and even potentially the underlying hardware. The provider meanwhile could choose only to monitoring physical resources or could monitor their users infrastructure.

GMonE is a recent monitoring tool which currently lacks any publicly available release. Despite this, GMonE's multi-layer monitoring solution and novel pubsub scheme are notable concepts which are not found in other contemporary monitoring tools.

Lattice is a monitoring platform for virtual resources which attempts to overcome the limitations of previous monitoring tools, including Ganglia and Nagios, that do not address elasticity, dynamism, and frequent change. Unlike other tools in this survey Lattice is not in itself a monitoring tool, rather it is a platform for developing monitoring tools. Lattice includes abstractions which are similar to other monitoring tools: producers, consumers, and probes. Additionally Lattice provides the concept of a data source and distribution framework. The data source is an producer which encapsulates the logic for controlling the collection and transmission of monitoring data in a series of probes which perform the actual data collection. The distribution framework is the mechanism which transmits a monitoring state between the producers and the consumers.

Lattice does not provide full implementations of these structures but rather provides building blocks from which third parties can develop full monitoring solution. This design is intended to allow developers to build monitoring solutions specific to their unique use cases. The separation of concerns between the various monitoring components allows components to be differently implemented and change over time without affecting the other components.

The notion of a framework for building monitoring tools is a novel break from the other tools surveyed in this paper. Conceptually Lattice can be used to build different monitoring tools for different use cases rather than reapplying existing tools to different use cases. While this allows for the best fitting tools possible it requires significant labor to develop and test tools based upon Lattice. Lattice does not provide a library of probes, requiring the developer to implement their library of data collection scripts, a significant limitation when compared to other tools including collected and Nagios. Additionally, Lattice requires the developer to make design decisions regarding the distribution framework; which network architectures, wire formats, discovery mechanisms, and so forth are used. This degree of effort is likely to be prohibitive to the vast majority of users.

Lattice being a framework for developing monitoring tools and not a tool in itself does not merit a direct classification and is capable of producing tools that fit every classification.

Varanus is a peer to peer monitoring tool designed for monitoring large scale cloud deployments. Designed to handle scale and elasticity, Varanus makes use of a k-nearest neighbor based group mechanism to dynamically group related VMs and form a layered gossip hierarchy for propagating monitoring state. Varanus uses three abstractions: groups, regions, and cloud over which the gossip layer of the hierarchy. Groups are collections of VMs assigned by Varanus' grouping strategy, regions are physical locations: data centers, cloud regions, or similar, and the cloud is the top-level abstraction including one or more regions. At the group and region level Varanus makes heavy use of the network to propagate state and state aggregates to related VMs. At the cloud level communication is slower and only periodic samples are propagated. This scheme attempts to exploit the inherent bandwidth differences in and between cloud regions. Heavy use of the network is used where bandwidth is unmetered and fast and reduced network usage occurs between regions where bandwidth is metered and limited.

Varanus has some points of commonality with Astrolabe and GEMS in that it utilizes a gossip hierarchy to propagate monitoring state but introduces applies these concepts in a manner more suitable to cloud monitoring.

3.3 Discussion

A main topic of this thesis is the discussion about the VMs/containers and the creation of them be it by the user's demand or that of a hacker. For that, we will give a short explanation of what VMs and containers are.

In computing, a virtual machine (VM) is the virtualization/emulation of a computer system. Virtual machines are based on computer architectures and provide the functionality of a physical computer. Their implementations may involve specialized hardware, software, or a combination.

As [9] suggest, Containers are an executable unit of software in which application code is packaged, along with its libraries and dependencies, in common ways so that it can be run anywhere, whether it be on desktop, traditional IT, or the cloud.

To do this, containers take advantage of a form of operating system (OS) virtualization in which features of the OS (in the case of the Linux kernel, namely the namespaces and cgroups primitives) are leveraged to both isolate processes and control the amount of CPU, memory, and disk that those processes have access to.

About containers. Cloud computing uses virtualization technologies to maximize the resource capacity utilization. One of the technology which is gladly used are containers, an alternative to VMs. The popularity of containers is rising accelerated, because of some advantages it provides. To make this clearer we point to a comparison in Figure 2, where a hypervisor and a container are used for application deployment. Hypervisors are preferred in deployment (a) when divergent operation system or divergent OS versions are required from applications on the same cloud. In contrast to that, containers are used in a system to make the sharing of an operating system possible for applications. With that, the size of these deployments can be remarkably smaller and thus leads to the possibility of storing hundreds of containers on a physical host. The usage of host OS in containers makes them very lightweight and has the benefit, that in case of a restarting, that can be done quickly and it is not necessary to reboot the OS. Because of the close integration of containers into the host operating system, the software overhead which is traced back to virtual machines, is reduced. However, the close integration also has disadvantages. It increases the attack area and security concerns are raising. In the case of a high-security environment to maintain isolation, containers might not be a good option. It is required to sacrifice the performance of an absolute-container deployment by introducing a VM to receive real isolation. The details regarding containers can also be found in [17] and [15]. The figure which describes the information about containers is also from [15].

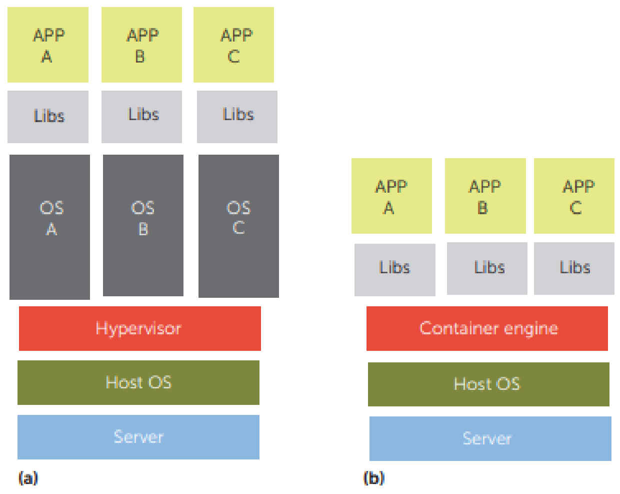


Figure 2: Two types of containers

An important representative who applies the concept of containers is Docker. Docker is an open-source software that addresses the isolation of applications using container virtualization. It uses the functionalities of the Linux kernel, such as cgroups and namespaces, to achieve the independent isolation of processes. The containers that Docker uses are created using base images. An image can consist of a preconfigured application stack or just contain the principles of the operating system. Docker uses a layered architecture for the arrangement. This means that each executed action forms a new level above the previous one. Docker files are responsible for the manual or automatic execution of commands. A Dockerfile can consist of various commands or instructions and arguments. These are listed one after the other to carry out actions for the creation of a new image on a base image. A simplification of the deployment process and the organization of deployment artifacts can be achieved with Docker files. In figure 3[15] we can observe possible layering combinations for application runtimes. PaaS here means platform as a service, which describes a service that provides a platform for developers of web applications via a cloud environment. A detailed description of Docker Container can be found in the referenced articles[15],[17].

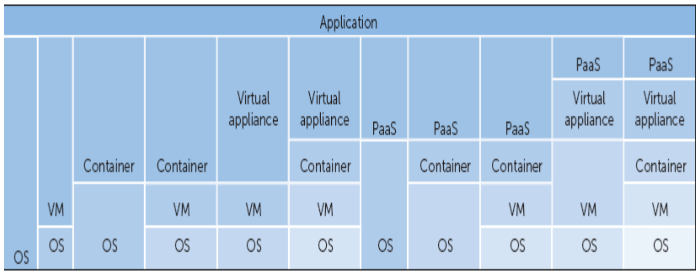


Figure 3: Possible layering combinations

4 RELATED WORK

There are not that many public works that focus on Cloud monitoring. Especially in [12], the authors say that Cloud monitoring is a very important thing that few talk about and it continues by giving a very detailed explanation of what Cloud monitoring is, basic concepts, open research issues and future directions.

Most of the related work we were able to found focus on the economic aspect of Cloud monitoring. In [13] the authors provide a very good comparison of commercial Cloud monitoring platforms. The tables that they provide give in the specified paper give examples and comparisons of key proprieties of commercial cloud monitoring platforms, key proprieties of open source cloud monitoring platforms and proprieties assessed by cloud performance and dependability monitoring services. The comparison focuses on the differences between scalability, elasticity, adaptability, timeliness, autonomicity, comprehensiveness, extensibility, intrusiveness, resilience, reliability, and availability and accuracy.

Another related work that we found was [14]. Here the authors also describe an overview of the commercial Cloud monitoring tools with some actual examples and comparisons of monitoring platforms.

5 CONCLUSION

In this paper we have discussed what Cloud monitoring means, while also addressing the problem of the lack of attention that it gets in public works. It can be concluded that monitoring plays an essential role in the management of cloud environments, which will continue to have an influential position in the future. To contextualize and study Cloud monitoring, we have provided background and definitions of key concepts. We also have listed some of the main platforms for Cloud monitoring that we were able to find and the applications and existing tools for it. Finally, we have discussed the questions given in the thesis topics, while also linking our paper with related works.

REFERENCES

[1] https://cloudinit.readthedocs.io/.
[2] https://datadoghq.com/.
[3] https://docs.sensu.io/.
[4] https://epsagon.com/.
[5] https://germainapm.com/.
[6] https://info.gartnerdigitalmarkets.com/.
[7] https://www.automai.com/.
[8] https://www.capterra.com/.
[9] https://www.ibm.com/cloud/learn/containers.
[10] https://www.sumologic.com/.
[11] https://www.whatsupgold.com/.
[12] Giuseppe Aceto, Alessio Botta, Walter De Donato, and Antonio Pescapè. Cloud monitoring: Definitions, issues and future directions. In *2012 IEEE 1st International Conference on Cloud Networking (CLOUDNET)*, pages 63–67. IEEE, 2012.
[13] Giuseppe Aceto, Alessio Botta, Walter De Donato, and Antonio Pescapè. Cloud monitoring: A survey. *Computer Networks*, 57(9):2093–2115, 2013.
[14] Khalid Alhamazani, Rajiv Ranjan, Karan Mitra, Fethi Rabhi, Prem Prakash Jayaraman, Samee Ullah Khan, Adnene Guabtni, and Vasudha Bhatnagar. An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art. *Computing*, 97(4):357–377, 2015.
[15] David Bernstein. Containers and cloud: From lxc to docker to kubernetes. *IEEE Cloud Computing*, 1(3):81–84, 2014.
[16] Stuart Clayman, Alex Galis, and Lefteris Mamatas. Monitoring virtual networks with lattice. In *2010 IEEE/IFIP Network Operations and Management Symposium Workshops*, pages 239–246. IEEE, 2010.
[17] Theo Combe, Antony Martin, and Roberto Di Pietro. To docker or not to docker: A security perspective. *IEEE Cloud Computing*, 3(5):54–62, 2016.
[18] Kaniz Fatema, Vincent C Emeakaroha, Philip D Healy, John P Morrison, and Theo Lynn. A survey of cloud monitoring tools: Taxonomy, capabilities and objectives. *Journal of Parallel and Distributed Computing*, 74(10):2918–2933, 2014.
[19] Benjamin König, JM Alcaraz Calero, and Johannes Kirschnick. Elastic monitoring framework for cloud infrastructures. *Iet Communications*, 6(10):1306–1315, 2012.
[20] HT Kung, Chit-Kwan Lin, and Dario Vlah. Cloudsense: Continuous fine-grain cloud monitoring with compressive sensing. In *HotCloud*, 2011.
[21] Jesús Montes, Alberto Sánchez, Bunjamin Memishi, María S Pérez, and Gabriel Antoniu. Gmone: A complete approach to cloud monitoring. *Future Generation Computer Systems*, 29(8):2026–2040, 2013.
[22] Javier Povedano-Molina, Jose M Lopez-Vega, Juan M Lopez-Soler, Antonio Corradi, and Luca Foschini. Dargos: A highly adaptable and scalable monitoring architecture for multi-tenant clouds. *Future Generation Computer Systems*, 29(8):2041–2056, 2013.
[23] Jonathan Stuart Ward and Adam Barker. Varanus: In situ monitoring for large scale cloud systems. In *2013 IEEE 5th International Conference on Cloud Computing Technology and Science*, volume 2, pages 341–344. IEEE, 2013.
[24] Jonathan Stuart Ward and Adam Barker. Observing the clouds: a survey and taxonomy of cloud monitoring. *Journal of Cloud Computing*, 3(1):1–30, 2014.