DOI 10.12694/scpe.v15i4.1055
Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 15, Number 4, pp. 345–355. http://www.scpe.org © 2014 SCPE

OPTIMIZING CLOUD RESOURCES ALLOCATION FOR AN INTERNET OF THINGS ARCHITECTURE[*]

BOGDAN MANAŢE[†], TEODOR-FLORIN FORTIŞ[‡], AND VIOREL NEGRU[§]

Abstract. The optimization of the cloud resources used to power a multi-agent Internet of Things architecture is an important issue which has an important impact on the overall operation cost of the architecture. The resources tenancy is a costly operation, thus their allocation and management should be optimized based on the usage patterns. The infrastructure for the multi-agent system should not be affected by the deployment or maintenance life cycle, operations require parts of the system, or even the entire system to be offline during the execution of scheduled procedures. This paper outlines of the importance of the infrastructure audit, which offers a good insight of how the resources are used, the geographical areas which are heavily used and where the allocation or release of used resources is mandatory. Also, the security audit, in a distributed multi-agent architecture that handles a large number of heterogeneous devices, represents a good mechanism for performance improvement.

Key words: Internet of Things, cloud computing, multi-agent systems

1. Introduction. The recent advances in Internet of Things (IoT) technologies and the cloud computing adoption have led to an increased usage of this two paradigms to create solid architectures that are able to handle hundred of thousands of concurrent connections, at the same time offering a good Quality of Service (QoS) and Quality of Experience (QoE) for the end user. The resources renting from public cloud providers for supporting the backbone infrastructure for an IoT architecture is beneficial for small and medium-sized enterprises (SME) or academic institutions which are trying to reach a large number of clients, because there is no need to upgrade or maintain the physical infrastructure. Even though the cloud client is not completely aware of the exact location of the hardware that delivers the required information, there are methods available that can determine the best route (considering both geographical location and bandwidth) for optimal performance [39].

For handling a big number of connections from a wide range of devices, an Internet of Things architecture should employ a fast, secure, reliable and fail safe infrastructure for the services offered to the end users which rely upon the manner in which the information is collected. Because of different usage patterns that result from the daily routine of different groups of users, the best solution for building an infrastructure for the IoT framework relies on the elasticity provided by the cloud computing paradigm [11, 15, 23, 33]. Also, the operational cost for maintaining the architecture up and running can be significantly reduced by releasing unused resources at daily time intervals when the audit operations report a low usage of the infrastructure.

From the reliability and operational standpoint, the core infrastructure for the multi-agent system needs to bypass any bottlenecks introduced by the on demand created infrastructure. Therefore, any replacement or restart of the virtual machines should be scheduled on isolated groups of virtual machines without affecting the system's response time.

Usually, on a non-complex application, the application access point and the database server are hosted on the same physical server, thus the business logic is very easy to manage. Therefore, the application access point has direct access to the required information without searching and querying any other network nodes that can be scattered in different data centers. In an Internet of Things framework, the business logic is separated on multiple physical servers, because it needs to process and store a large amount of information sent by the clients. Considering the tremendous amount of data that needs to be stored, the database instances are separated on multiple virtual machines, so that the database instances can be grouped in clusters for a better management and distribution of stored data. Given the heterogeneous nature of the information handled by the Internet of Things architecture the data that needs to be stored can be also grouped based on its type, therefore various

[*] EXTENDED VERSION OF [18]

[†] West University of Timişoara, Faculty of Mathematics and Informatics, Computer Science Department, bvd. V. Pârvan, 4, 300223,Timişoara, Romania(bogdan.manate@info.uvt.ro).

[‡] Institute e-Austria Timişoara & West University of Timişoara, Faculty of Mathematics and Informatics, Computer Science Department, bvd. V. Pârvan, 4, 300223,Timişoara, Romania(fortis@info.uvt.ro).

[§] Institute e-Austria Timişoara & West University of Timişoara, Faculty of Mathematics and Informatics, Computer Science Department, bvd. V. Pârvan, 4, 300223,Timişoara, Romania(vnegru@info.uvt.ro).

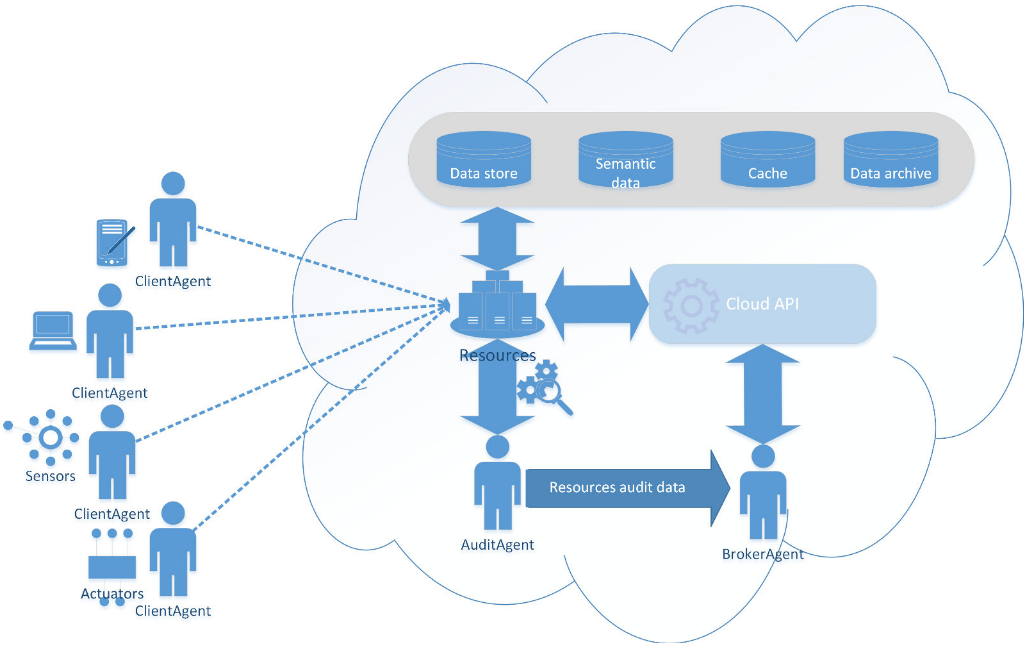

Fig. 1.1. *The actors that interact with the multi-agent architecture [18].*

types of storage solutions can be used based on the information type. The information which is collected on a regular basis from the sensors can be stored in a time series database, the semantic information about devices can be stored using a triple store database and the information about the system's users (e.g. credentials and preferences) can be stored in a document based database.

The same need for intensive infrastructure management is identified in social networks like Facebook [8], Twitter [12], LinkedIn [28] and MySpace [7] and online video streaming providers like Netflix [41] and Hulu [40], where data integrity and services up time play an important role because the services downtime may generate significant revenue losses.

This papers presents how an on demand infrastructure is created to support a multi-agent architecture which operates in the Internet of Things context and how the resources allocation can be optimized based on the usage patterns inferred from the audit operations.

This paper is structured as follows: in Section 2 is presented a literature review of existing infrastructures, in Section 3 is presented the multi-agent architecture designed for Internet of Things governance that needs to manage the underlying cloud infrastructure, in Section 4 is presented a solution for a cloud infrastructure management able to scale an IoT architecture, in Section 5 it is emphasized the importance of infrastructure audit and underline the benefits of a proper infrastructure audit, in Section 6 are presented two methods that enhance the resources utilization and the system scalability and finally, in Section 7 are presented the final conclusions of this paper.

2. Background. By unifying the Internet of Things, Cloud Computing and network edge services a new paradigm emerges called Fog Computing [4], which offers compute power, support for data storage and provides necessary infrastructure for connecting the sensors and clouds. The researchers from CISCO have proposed the Fog Computing term to exemplify a system that is characterize by low latency, ample geographical distribution, location awareness, mobility and heterogeneity. In the context of Fog Computing the services are closer to the consumer delivering this way the required information in a fast and reliable manner. The support for bi-directional data flow (from devices to the cloud and conversely) allows the system to have a greater control over the data sources and the cloud resources. In case of a data source failure, the system can isolate the affected data sources and any important information is inferred from the neighboring devices based on the localization service. Also, the bi-directional data flow enables the multi-agent system to send commands to devices that are

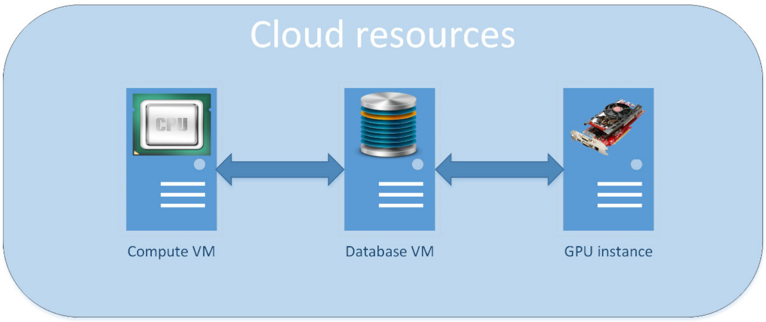

Fig. 2.1. *The types of virtual machines used in the IoT infrastructure [18].*

able to execute simple or complex operations (like actuators).

Beside moving some of the logic to the edge of the network it is imperative to take into account the geographical distribution of the resources that need these services [22]. The services distribution depends on the audit performed on that parts of the network characterized by same location and the number of clients that the system should service. The services are distributed based on the number of possible clients in a given geographical area, therefore a big number of resources will be available for urban areas to service the existing clients and a smaller number of resources will be distributed for the users living in the vicinity of the urban areas. In the edge computing (EC) an important role is played by context awareness, so that a resource it is aware of its current location, the surrounding resources and where is the case the resource can be aware of the entire surrounding context. Thus the methods for information aggregation can benefit from the informations exchanged by the edge services about the surrounding context.

Even though the services are brought as close as possible to the end-users, parts of the data requested by the users is sometimes stored on different database servers situated in various data centers. In [16] is proposed a solution based on the data replication mechanism provided by distributed databases. The important data sets stored in the main data center are replicated on cheap commodity hardware situated at the edge of the network closer to the edge services, reducing by this way the network latency and core network utilization. Also, some of the information can be stored for a small amount of time on the client devices, enabling this way a peer-to-peer information exchange between different users.

By utilizing the multi-agent system based on a distributed infrastructure the end-users should benefit from a good quality of service (QoS) enhanced by a pleasant experience which respects high standards of quality of experience (QoE) [24]. The QoE term is usually used to express the overall experience of using a multimedia system. However, considering the multitude of devices that are encompassed by the Internet of Things framework it is worth noting that these devices are used to improve the user's experience in an environment.

In order to scale the multi-agent architecture proposed in [19], a large number of virtual machine instances are launched in the cloud so that the system load can be evenly distributed on the available resources. The system can be scaled as long as the cloud provider has the necessary resources available, otherwise additional resources may be launched using another public cloud provider or a private cloud. A solution for this problem is offered by the open-source platform mOSAIC [27], which offers an unified application programming interface and a platform for developing large scale applications that are using resources rented from multiple cloud providers. The resources provisioning mechanism is based on a multi-agent architecture (CloudAgency) [38], that enables the platform to rent the best cost effective heterogeneous resources from different cloud providers based on a service level agreement (SLA) provided by the user. The best solution is matched by a semantic reasoning module embeded in the multi-agent system that operates on a Cloud Ontology.

Aneka [37] is another a cloud computing platform that uses cloud resources from the Microsoft Azure cloud for creating services oriented applications. The Aneka platform offers a configurable services container, services discovery and load balancing, therefore the developers can focus on the application development and less on the infrastructure management.

Given the fact that it is impossible for a single cloud provider to scatter its data centers around the globe to

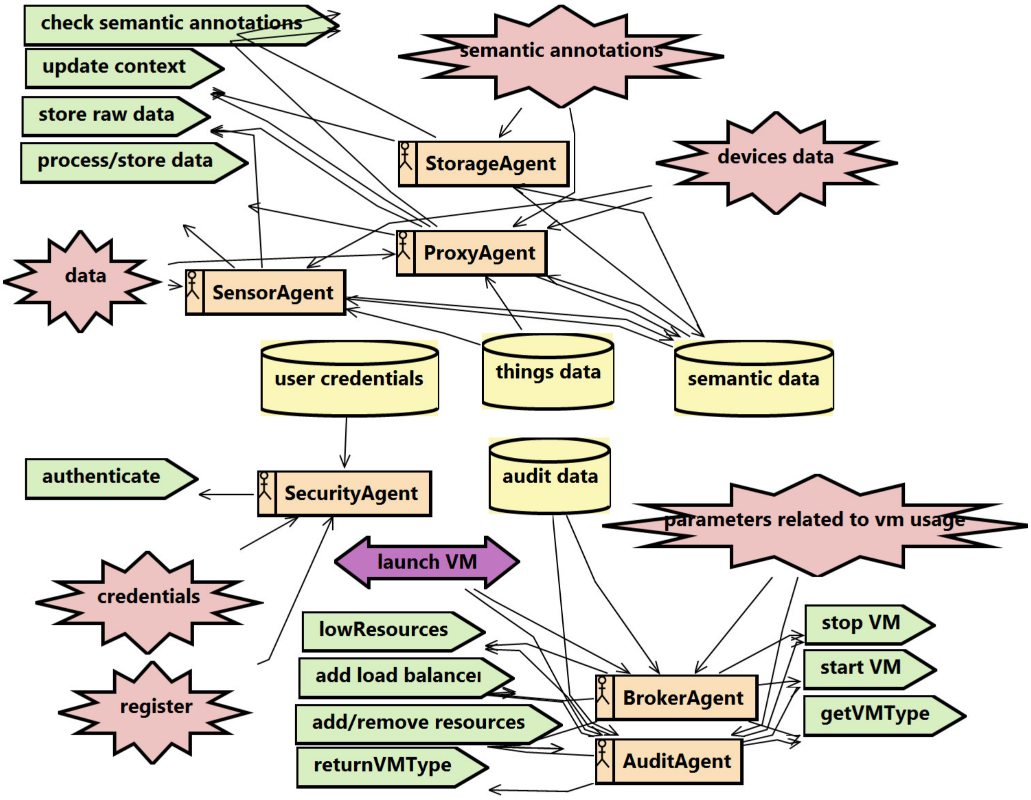

FIG. 3.1. *The multi-agent architecture [19]*

offer on demand computing power as close as possible to their clients, a viable solution to overcome this problem is represented by a cloud federation [5]. By using a federation of clouds and a location discovery services, the resources are rented from the closest available cloud provider thus enabling the infrastructure to offer fast and localized support.

3. Architecture. The proposed architecture for the multi-agent system [19] presented in Figure 3.1 uses the agents as independent and mobile entities, which are specialized to solve specific domain oriented issues that are introduced by the Internet of Things paradigm. As seen in Fig. 1.1, groups of agents are working together to gather, annotate, process and store data in a cloud environment. Other collections of specialized agents are distributed on client devices based on the devices semantic definition. By having the agents distributed both on the client devices and on the cloud infrastructure, the communication and the management of the agents is delegated to the core implementation of the multi-agent platform, thus the resulting system operates as an integrated environment.

The information, which may be sent as partially processed or raw data, flows from the clients to the cloud endpoints that are managed by the instances of the ProxyAgent, which are routing the information based on its type to the available resources. When the information reaches one of the end-points of the cloud, it can be semantically annotated and sent for further processing. Considering the special cases when the context aware component is active, the information related to a particular context can be used for updating the current context if the information received contains relevant values that depict a modification of the actual state. The information that contributes to context modelling is managed by ContexManagerAgent [21], therefore the ProxyAgent needs to route the information related to the context to a virtual machine where the ContextManagerAgent is instantiated.

The data cache plays an important role in a system that handles massive quantities of information because it has a big impact on the system performance. The most accessed information is stored in the RAM memory, so that all requests trying to read the information from a database are routed directly to the information stored by the cache system. The agents that are handling a large amount of information are developed with an of the shelf local caching system to reduce the time needed for information access.

When data gathered from the devices is older than a specified amount of time it is transferred to the data archive database. The archive database compress and stores the aggregated information in order to facilitate a later information retrieval of the archived data. When the information stored in the archive database becomes obsolete the data archive is subjected to a purge operation that aims to remove the existing information which has not been used in a long time.

The multi-agent system was developed to serve a general purpose when it comes to data collection and processing, but it can be used in various domains like ambient intelligence, smart city management, ambient assisted living and for supervising the industrial processes.

4. Infrastructure management.

The proposed multi-agent system employs three types of virtual machines, which have to accomplish a specific task in the architecture:

- Compute VM - compute optimized instances which offer the highest performing processors. These types of instances are used for the data collection end-points. The multi-agent container runs on these instances, where agents specialized for data processing and semantic annotation are instantiated.
- Database VM - storage optimized instances that are able to offer support for memory-intensive and random I/O operations. The instances of this type can be grouped into clusters depending on the technical specifications provided by the company which delivers the database software solution. A virtualization advisor [34] can be used to determine based on the audit data which is the best configuration for the database instances.
- GPU instances - instances that have attached a graphical and general purpose GPU. This type of instance is used for data mining [17], data-matching [29], intrusion detection [14, 36] and learning [10].

4.1. Managing compute instances.

In the paper [20] it is presented a suite of benchmark tests performed on a prototypical implementation of the Internet of Things architecture briefly presented in Section 3. The benchmarks results define the maximum number of connections that can be handled by different types of virtual machines rented from Amazon EC2. The maximum threshold for every virtual machine type can be used by the BrokerAgent, as a trigger, to launch a new instance in order to distribute the system load.

The standard virtual machine images, that are built with the purpose of supporting the architecture, are bundled with necessary software packages to start the architecture. When a non standard image is launched from the cloud provider repository, the BrokerAgent handles the deployment phase as well the registering phase after which the image is marked as *ready* to process the data collected from the devices.

Event though the implementation of the proposed multi-agent architecture targets the Scala programming language and Akka toolkit, to benefit from the highly concurrent and distributed nature of the proposed multi-agent system, the out of the box scheduling mechanism [3] implemented by Scala/Akka software stack can be greatly improved with the addition of load balancers [32], which are offered by default by the majority of the cloud providers. Therefore, the BrokerAgent is designed to handle this scenario by grouping the targeted virtual machines, automatically, requesting a new load balancer for the grouped instances from the cloud provider using the cloud provider's public API.

Another benefit resulted from the usage of a multi-agent system on top of the Akka toolkit is the standard implementation of the communication protocol, which is the default option supported by every agent. This way the agents are able to exchange information regardless of their location, hence, the infrastructure might be exposed to a higher network traffic when the actors are deployed on a virtual machine that has been launched in a different data centers.

The audit operations performed by the AuditAgent on the entire infrastructure are important, because they offer valuable information about the resources utilization, unauthorized access attempts, system loading, the number of newly added/removed devices and the geographical distribution of the end users. In order to offer edge computing services for the clients the system should be aware of the resources' positions which are useful when a local system failure occurs. By knowing the exact location of the resources and the location of the clients, the edge services can be dynamically managed and deployed offering a flexible alternative when a local resource becomes unavailable.

4.2. Managing database instances. The instance type that is hosting the database software solutions represents an important instance type for the architecture, because it assures the data persistence. The database instances need to be configured based on the technical specifications of the database software which is launched on these instances. The special situation that arise when database instances are grouped together in a cluster or ring require additional instructions to join the group after a restart. Also, other tuning operations can be executed to improve the database response time. Therefore, the BrokerAgent has to mitigate the interaction between the cluster and the new database instance and any other operations that require interaction with the database software. For applications targeting the Java platform, the JSch [1] library can be used to send commands over SSH to a virtual machine in a secure manner.

The operations executed via the command line interpreter, like joining a cluster, gathering audit data or a force data replication, are executed by any of the agents running on the local agents container or running on a remote virtual machine, thus allowing the multi-agent system to have total control over the launched instances. Therefore, for security reasons, the interactions with the underlying operating system should be executed only by the agents which have a very well defined role in the infrastructure management like AuditAgent or BrokerAgent.

Some NoSQL databases, like Riak, recommend the usage of a load balancer [1] as a best practice, because every node that is a member of a cluster is able to handle the incoming requests, hence the incoming requests can be routed to any available instance.

4.3. Managing GPU instances. The GPU instances are offered at high rates, because the GPU boards are bundled with expensive hardware. Hence, only a handful of cloud providers offer such instances, some of them are big players in cloud computing domain like Amazon, Nimbix, Peer1, Penguin computing and Softlayer.

The data processing using a high performance GPU is very fast, hence the data traffic between the GPU instances and the data store can simply overcome the infrastructure network capability. To solve this problem, the GPU instances need to be started in the same data center where the data which is subject to processing is available. Also, the network connection between the database instances and the GPU instances should offers support for a higher bandwidth to speed up the data transfer. A better solution for networking is a cluster network where the instances launched in the same cluster group are started on the same physical server rack, so that the cluster network provides high network bandwidth and low latency for data transfer between instances.

The agents of the proposed multi-agent system that are running tasks on the GPU instances are implemented using ScalaCL [2] a library that lets the programmers to run Scala code on the GPUs in a very natural way. The tasks distribution is handled by a router agent, which uses a Round Robing algorithm to distribute the information stored in the local mailbox (an internal queue).

5. Infrastructure audit. The system audit is important in an environment where tens of thousands of virtual machines represent the backbone of a multi-agent architecture (Fig. 5.1). The audit is useful for automatic maintenance as well as for human operators, so that the data gathered by the audit operation can be used to maximize the infrastructure performance and to reduce the operational costs. The audit operations offer important data about the network latency, CPU usage, GPU usage, RAM memory loading, and it can also verify if the resources rented from the cloud provider respect the service level agreement (SLA) [25], which is very important for the infrastructure stability. The multi-agent system relies on the cloud infrastructure to operate at best performance parameters, so that any change in the infrastructure components can affect the system's overall performance. Thus, the audit operations must be scheduled after every start/restart of a virtual machine to validate the changes [6].

Because the multi-agent architecture proposed in [19] was developed to target a wide-spread geographical area, the audit information is useful to determine which area offers a high QoS/QoE for end-users. Thus, the edge computing services offered for certain geographical areas can be configured for best performance results considering the audit data collected for the specified areas.

The AuditAgent uses a database to store different audit results based on the execution time. Storing audit results for a long period of time is useful to identify the parts of the infrastructure that are extensively used so

[1] http://www.jcraft.com/jsch/

[2] https://code.google.com/p/scalacl/

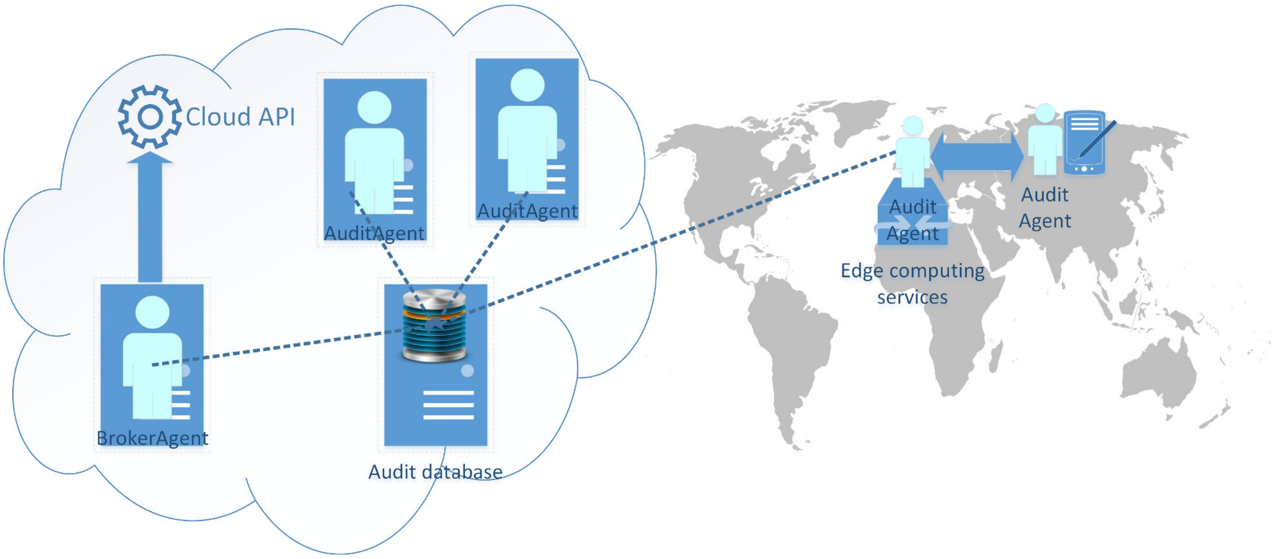

FIG. 5.1. *The IoT infrastructure audit [18].*

that further actions may be taken to improve performances by moving some of the resources situated in areas with low traffic to the area where the infrastructure is experiencing peaks of traffic.

Considering the vast applications of the Internet of Things in domains like health care, smart city, ambient intelligence and industrial, a lot of private information is transferred between agents which are situated either on the client side or in the cloud. To protect the private information of the end-users it is imperative to execute security audit operations [35] over different components of the system. The heterogeneous character of the devices that interact in the IoT framework offer a wide range of possibilities for a cyber-attack, therefore periodic audit operations are compulsory. The security audit needs to target the client-side applications, edge computing services and the cloud infrastructure periodically, so that any attempt or unauthorized usage of the system has to be immediately identified and reported.

Another important aspect of the infrastructure audit is related to the detection of the intrusion attempts. The detection of intrusion attempts can be automatically handled by the SecurityAgent, which has a set of rules implemented to deal with such situations. Real time intrusion detection for the Internet of Things paradigm has been implemented in SVELTE [31] project with an impressive detection rate of almost 100%, thus the task of detecting potential attackers can be assigned to an agent which has full access to the data exchange inside of the multi-agent system.

6. Resources usage optimization. Even though there are many research papers [2, 9, 13, 26, 30] focused on the optimization of the cloud resources usage, for the particular case regarding the proposed multi-agent architecture the resource usage can be optimized using a two-phase method base on resources pooling and virtual machine pre-warm-up. In order to optimize the resources usage in an Internet of Things architecture there should be a balance between assuring the system stability and resources usage. The first method presented bellow is based on a pool of resources and is employed when no information about the system usage is available. After the audit operations manage to gather a significant amount of data about how and when the system is accessed by the end users the multi-agent architecture will be able to pre-warm the virtual machines just in time for the expected traffic peak.

6.1. Resources pooling. The resource pooling method is used when there is no audit data available about the system's usage so that the virtual machines cannot be prepared for the network traffic peaks. A variable number of compute virtual machines are provisioned with the required software stack and are kept in a resources pool for when the system is under heavy load. As seen in Fig. 6.1 when the system is under heavy load the available resources are moved from the resources pool and attached to the load balancer that is

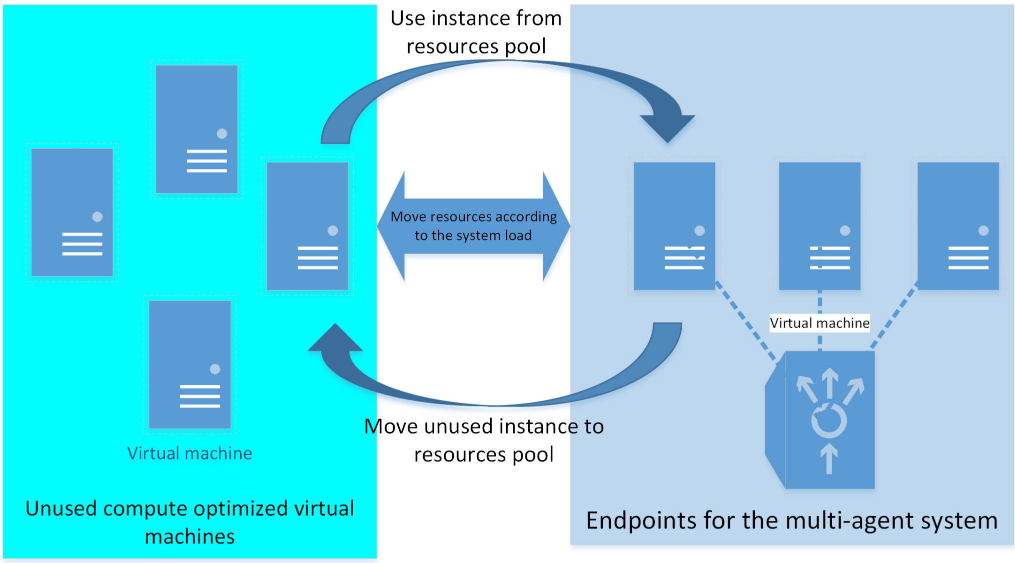
Use instance from resources pool
Move resources according to the system load
Virtual machine
Virtual machine
Move unused instance to resources pool
Unused compute optimized virtual machines
Endpoints for the multi-agent system

Fig. 6.1. *Resources pool.*

experiencing heavy network traffic.

Keeping the underutilized resources running is the main drawback of this method but is a fair compromise for keeping the system running in special situations generated by massive network traffic.

6.2. Virtual machines 'warm-up'. The Internet of Things is characterized by highly localized and repetitive events, therefore after collecting enough information about the system usage it is very easy to forecast where and when are required extra cloud resources. The information about the system usage can be extracted by analyzing the audit data collected by the AuditAgent. The number of virtual machines present in the resources pool can be gradually decreased as the information gathered from the audit operations has relevant information about the usage time frame.

In Fig. 6.2 is presented a general usage sample of the available resources. The point B represents the maximum number of connections that can be handled by the multi-agent system using the existing resources. The point C represents a forecast of the maximum number of connections for the current time interval. In order to avoid an overloading of the system by reaching the critical point B, when no additional virtual machines are started to take over the excess traffic, the point A was selected as the best time when a new virtual machine should be started. The best time is calculated by subtracting the time needed for the virtual machine to start (the time needed for booting and for installing the required software stack) from the time when the critical point B will be reached.

7. Conclusion and future work. This paper presents a method for a cloud infrastructure management and cloud resources allocation based on a multi-agent solution. The proposed solutions aim to overcome the issues encountered when managing an on-demand cloud infrastructure for such a dynamic framework like the Internet of Things. Because the entire logic for infrastructure management is detached to specialized agents, that operate on data collected from the system logs or from user defined constraints, it can be used as a standalone component in other contexts, that have similar requirements like the multi-agent system used for Internet of Things governance and which are required to operate using the infrastructure as a service paradigm.

Another important aspect underlined in this paper is related to infrastructure audit and the positive impact on the system performance when the data collected during the audit operation is used to dynamically reconfigure the system. As presented in Section 6 the audit operation also play an important role for resources usage optimization. The audit information is useful when dealing with location aware services, because it offers valuable information about the current state of the machines which are hosting the edge network services.

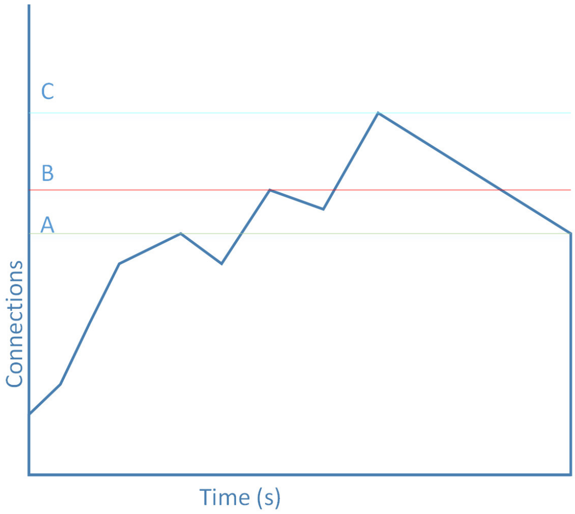

Fig. 6.2. *A general usage sample*

Hence, the regions which are characterized by heavy network traffic can benefit from a new set of computing resources during traffic peak.

For further research it is planned to develop a feature of the multi-agent system that analyses the instances usage patterns related to users profiles, so that the agents will be able to significantly reduce the number of the instances launched in the resources pool. Therefore the multi-agent system will provide just in time the instances needed for balancing the incoming requests, thus reducing the unnecessary costs generated by the idle instances from the resources pool.

Acknowledgment. The work of the first author was partially supported by the strategic grant POSDRU/159/1.5/S/137750, "Project Doctoral and Postdoctoral programs support for increased competitiveness in Exact Sciences research" cofinanced by the European Social Fund within the Sectoral Operational Programme Human Resources Development 2007–2013 and Romanian Government grant PN-II-ID-PCE-2011-3-0260 (AMICAS). The views expressed in this paper do not necessarily reflect those of the corresponding projects consortium members.

REFERENCES

[1] *Riak load balancing.* `http://docs.basho.com/riak/1.3.1/cookbooks/Load-Balancing-and-Proxy-Configuration/`. Accessed: 2014-09-08.

[2] V. Aggarwal, V. Gopalakrishnan, R. Jana, K. Ramakrishnan, and V. A. Vaishampayan, *Optimizing cloud resources for delivering IPTV services through virtualization*, in Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on, IEEE, 2012, pp. 1–10.

[3] M. Bevilacqua-Linn, M. Byron, P. Cline, J. Moore, and S. Muir, *Sirius: distributing and coordinating application reference data*, in Proceedings of the 2014 USENIX conference on USENIX Annual Technical Conference, USENIX Association, 2014, pp. 293–304.

[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, *Fog computing and its role in the internet of things*, in Proceedings of the first edition of the MCC workshop on Mobile cloud computing, ACM, 2012, pp. 13–16.

[5] R. Buyya, R. Ranjan, and R. N. Calheiros, *Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services*, in Algorithms and architectures for parallel processing, Springer, 2010, pp. 13–31.

[6] F. Doelitzscher, C. Fischer, D. Moskal, C. Reich, M. Knahl, and N. Clarke, *Validating cloud infrastructure changes by cloud audits*, in Services (SERVICES), 2012 IEEE Eighth World Congress on, IEEE, 2012, pp. 377–384.

[7] M. Farkas, *Going where patrons are: Outreach in MySpace and Facebook*, American Libraries, 38 (2007), p. 27.

[8] N. Farrington and A. Andreyev, *Facebooks Data Center Network Architecture*, in IEEE Opt. Interconnects Conf, Citeseer, 2013, pp. 5–7.

[9] J. M. Ferris, *Methods and systems for optimizing resource usage for cloud-based networks*, Aug. 22 2008. US Patent App. 12/196,459.

[10] L. F. Gruber and M. West, *GPU-Accelerated Bayesian Learning and Forecasting in Simultaneous Graphical Dynamic Linear Models*, tech. report, Technical report, Department of Statistical Science, Duke University, 2014.

[11] D. Guinard, C. Floerkemeier, and S. Sarma, *Cloud computing, REST and mashups to simplify RFID application development and deployment*, in Proceedings of the Second International Workshop on Web of Things, ACM, 2011, p. 9.

[12] T. Hoff, *Scaling Twitter: Making Twitter 10000 Percent Faster*, High Scalability, (2009).

[13] S. Khatua, A. Ghosh, and N. Mukherjee, *Optimizing the utilization of virtual resources in Cloud environment*, in Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS), 2010 IEEE International Conference on, IEEE, 2010, pp. 82–87.

[14] S.-i. Kim, W. Edmonds, and N. Nwanze, *On GPU accelerated tuning for a payload anomaly-based network intrusion detection scheme*, in Proceedings of the 9th Annual Cyber and Information Security Research Conference, ACM, 2014, pp. 1–4.

[15] M. Kovatsch, S. Mayer, and B. Ostermaier, *Moving application logic from the firmware to the cloud: Towards the thin server architecture for the internet of things*, in Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on, IEEE, 2012, pp. 751–756.

[16] Y. Lin, B. Kemme, M. Patino-Martinez, and R. Jimenez-Peris, *Enhancing edge computing with database replication*, in Reliable Distributed Systems, 2007. SRDS 2007. 26th IEEE International Symposium on, IEEE, 2007, pp. 45–54.

[17] W. Ma and G. Agrawal, *A translation system for enabling data mining applications on GPUs*, in Proceedings of the 23rd international conference on Supercomputing, ACM, 2009, pp. 400–409.

[18] B. Manaţe, T.-F. Fortiş, and V. Negru, *Infrastructure Management Support in a Multi-Agent Architecture for Internet of Things*, in European Modelling Symposium EMS2014 (EMS2014), Pisa, Italy, Oct. 2014.

[19] B. Manaţe, T.-F. Fortis, and P. Moore, *Applying the Prometheus methodology for an Internet of Things architecture*, in Utility and Cloud Computing (UCC), 2014 Fourth IEEE International Conference on Utility and Cloud Computing, IEEE, 2014.

[20] B. Manaţe, V. I. Munteanu, and T.-F. Fortiş, *Towards a Scalable Multi-agent Architecture for Managing IoT Data*, in P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on, 2013, pp. 270–275.

[21] B. Manaţe, V. I. Munteanu, T. F. Fortis, and P. T. Moore, *An intelligent context-aware decision-support system oriented towards healthcare support*, Complex, Intelligent and Software Intensive Systems (CISIS), 2014 Eighth International Conference, (2014), pp. 386–391.

[22] R. Manning, *Dynamic and distributed managed edge computing (MEC) framework*, May 20 2004. US Patent App. 10/850,291.

[23] N. Mitton, S. Papavassiliou, A. Puliafito, and K. S. Trivedi, *Combining Cloud and sensors in a smart city environment*, EURASIP Journal on Wireless Communications and Networking, 2012 (2012), pp. 1–10.

[24] R. F. Moghaddam and M. Cheriet, *A Note on Quality of Experience (QoE) beyond Quality of Service (QoS) as the Baseline*, arXiv preprint arXiv:1407.5527, (2014).

[25] D. Ouelhadj, J. Garibaldi, J. MacLaren, R. Sakellariou, and K. Krishnakumar, *A multi-agent infrastructure and a service level agreement negotiation protocol for robust scheduling in grid computing*, in Advances in Grid Computing-EGC 2005, Springer, 2005, pp. 651–660.

[26] N. Paton, M. A. De Aragão, K. Lee, A. A. Fernandes, and R. Sakellariou, *Optimizing utility in cloud computing through autonomic workload execution*, Bulletin of the Technical Committee on Data Engineering, 32 (2009), pp. 51–58.

[27] D. Petcu, B. Di Martino, S. Venticinque, M. Rak, T. Máhr, G. E. Lopez, F. Brito, R. Cossu, M. Stopar, S. Šperka, et al., *Experiences in building a mOSAIC of clouds*, Journal of Cloud Computing, 2 (2013), pp. 1–22.

[28] J. M. Pujol, G. Siganos, V. Erramilli, and P. Rodriguez, *Scaling online social networks without pains*, in Proc of NETDB, 2009.

[29] C.-P. Pungila, M. Reja, and V. Negru, *Efficient parallel automata construction for hybrid resource-impelled data-matching*, Future Generation Computer Systems, 36 (2014), pp. 31–41.

[30] J. Rao, X. Bu, C.-Z. Xu, and K. Wang, *A distributed self-learning approach for elastic provisioning of virtualized cloud resources*, in Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2011 IEEE 19th International Symposium on, IEEE, 2011, pp. 45–54.

[31] S. Raza, L. Wallgren, and T. Voigt, *SVELTE: Real-time intrusion detection in the Internet of Things*, Ad hoc networks, 11 (2013), pp. 2661–2674.

[32] M. Sharma, Y. Anitha, and P. Sharma, *An Optimistic Approach for Load Balancing in Cloud Computing*, (2014).

[33] J. Soldatos, M. Serrano, and M. Hauswirth, *Convergence of utility computing with the internet-of-things*, in Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on, IEEE, 2012, pp. 874–879.

[34] A. A. Soror, U. F. Minhas, A. Aboulnaga, K. Salem, P. Kokosielis, and S. Kamath, *Automatic virtual machine configuration for database workloads*, ACM Transactions on Database Systems (TODS), 35 (2010), p. 7.

[35] X. Sun and C. Wang, *The Research of Security Technology in the Internet of Things*, in Advances in Computer Science, Intelligent System and Environment, Springer, 2011, pp. 113–119.

[36] G. Vasiliadis, S. Antonatos, M. Polychronakis, E. P. Markatos, and S. Ioannidis, *Gnort: High performance network intrusion detection using graphics processors*, in Recent Advances in Intrusion Detection, Springer, 2008, pp. 116–134.

[37] C. Vecchiola, X. Chu, and R. Buyya, *Aneka: a software platform for .NET-based cloud computing*, High Speed and Large Scale Scientific Computing, (2009), pp. 267–295.

[38] S. Venticinque, R. Aversa, B. Di Martino, and D. Petcu, *Agent based Cloud Provisioning and Management-Design and Prototypal Implementation.*, in CLOSER, 2011, pp. 184–191.

[39] B. Wickremasinghe, R. N. Calheiros, and R. Buyya, *Cloudanalyst: A cloudsim-based visual modeller for analysing cloud*

computing environments and applications, in Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, IEEE, 2010, pp. 446–452.

[40] F. Wu, W.-H. Chen, P. R. Graham, and G. D. Pelton, *Efficiently distributing video using a hybrid network that uses existing infrastructure*, Oct. 13 2009. US Patent 7,602,846.

[41] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, *Large-scale parallel collaborative filtering for the netflix prize*, in Algorithmic Aspects in Information and Management, Springer, 2008, pp. 337–348.

Edited by: Dana Petcu
Received: November 15, 2014
Accepted: January 15, 2015