A Pattern for Fog Computing

MADIHA H. SYED, Florida Atlantic University
EDUARDO B. FERNANDEZ, Florida Atlantic University
MOHAMMAD ILYAS, Florida Atlantic University

Fog Computing is a new variety of the cloud computing paradigm that brings virtualized cloud services to the edge of the network to control the devices in the IoT. We present a pattern for fog computing which describes its architecture, including its computing, storage and networking services. Fog computing is implemented as an intermediate platform between end devices and cloud computing data centers. The recent popularity of the Internet of Things (IoT) has made fog computing a necessity to handle a variety of devices. It has been recognized as an important platform to provide efficient, location aware, close to the edge, cloud services. Our model includes most of the functionality found in current fog architectures.

Keywords: Software architecture, Cloud computing, Patterns, Security

Additional Key Words and Phrases: Fog computing, edge computing, Internet of Things, architecture patterns

ACM Reference Format:

Madiha Syed, Eduardo B. Fernandez and Mohammad Ilyas. 2016. A Pattern for Fog Computing. *ACM Proc. 10th Travelling Conference on Patterns Language of Programming. VikingPLoP* (April 2016), 10 pages.
DOI: http://dx.doi.org/10.1145/3022636.3022649

1. INTRODUCTION

Cloud Computing has established its popularity by providing on-demand computing services to a large and diverse set of users and systems. It has led to a global shift in the computing world and the paradigm itself is evolving as new functions or technologies become available. Intelligent and interactive environments like Internet of Things (IoT) have found application in various domains. Billions of smart devices are connected to the internet and are producing huge amounts of data. These systems can only be utilized to their full potential if their complexity is handled properly. IoT has made the limitations of clouds more apparent as these systems typically require low latency, support for heterogeneity, mobility, geographical distribution, location awareness, etc. The traditional Cloud is not enough to satisfy the device management and data processing requirements of IoT. Cloud service providers like Amazon, IBM, Cisco, etc., are offering solutions to cater to IoT systems. Fog computing provides one such solution where fog is defined as a collection of numerous distributed tiny clouds deployed closer to the devices at edge of the network.

We present here a pattern for Fog Computing. Our audience includes system architects, designers, and software developers, as well as others who are interested in building fog computing solutions. In addition, our pattern can also help users who want to understand the components involved and how they relate to each other. We use the POSA (Pattern-oriented Software Architecture) template to describe this pattern [Buschmann et al. 1996]. A pattern is a solution to a recurring problem in a specific context. A Reference Architecture (RA), mentioned later in the paper, is a standardized, generic

Author's address: Madiha H. Syed, email: msyed2014@fau.edu; Eduardo B. Fernandez, email: ed@cse.fau.edu; Mohammad Ilyas, email: ilyas@fau.edu; Dept. of Computer and Elect. Eng. and Computer Science Florida Atlantic University, Boca Raton, FL 33431, USA.

VikingPLoP '16, April 07-10, 2016, Leerdam, AA, Netherlands
© 2016 ACM. ISBN 978-1-4503-4200-1/16/04$15.00
DOI: http://dx.doi.org/10.1145/3022636.3022649

software architecture with no platform dependencies, valid for a particular domain. An RA can be built of patterns and is also considered itself as a pattern.

Fog Computing is an important addition to cloud ecosystems. An ecosystem is the expansion of a software product line architecture to include systems outside the product which interact with the product [Bosch 2009]. In an attempt to precisely model a cloud ecosystem, we describe this ecosystem in the form of a pattern diagram. Each of the components of the ecosystem can be represented as patterns and RAs using UML models. Figure 1 shows a partial cloud ecosystem where the Cloud RA is the core pattern and Cloud Security RA (SRA) adds security patterns to it [Fernandez et al. 2015]. A Cloud Compliance RA includes patterns that describe how regulations apply to the Cloud RA. Cloud IaaS, PaaS and SaaS are service layers of the Cloud. Association between the components in the pattern diagram can represent derived (specializations) patterns, for example, Cloud SRA is derived from Cloud RA, or by simple association indicating contribution of one pattern to the other. Further elaboration about each of these components and more comprehensive cloud ecosystem diagram can be found in [Fernandez et al. 2016].

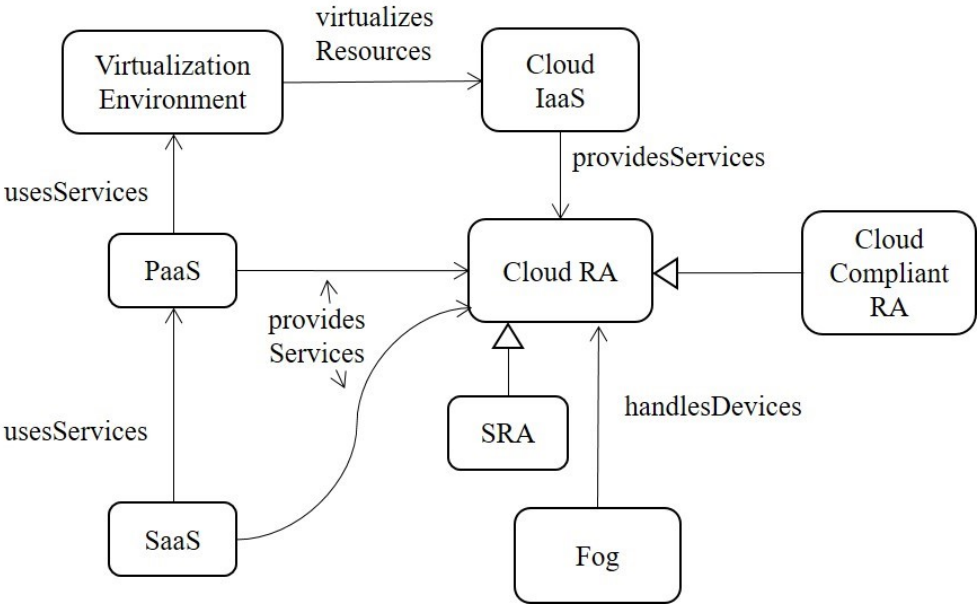

Fig. 1. A partial cloud ecosystem

2. FOG COMPUTING PATTERN

2.1 AKA

Fog networking, Fog, Fogging, Edge of network computing, Edge Computing, Mobile-edge computing

2.2 Intent

Fog Computing is a virtualized platform that stands between cloud computing systems and Internet devices, providing to these computation, storage, and networking services and allowing a cloud to control and communicate with these devices and to the devices to perform some functions in the fog or the cloud.

2.3 Example

Smartville has a Smart Traffic Control System. It is comprised of many heterogeneous devices like traffic lights, vehicles, emergency vehicles, and pedestrians with mobile devices. Each device has a diverse set of resources and data. The number of devices that are connected to the Internet is growing rapidly. The devices are becoming smarter and the need for more applications is ever increasing. We are finding it difficult to manage and coordinate these devices. Service and application requests from the cloud need to be processed in real time to control these devices. With a large number of devices the volume of data produced is huge. It is a challenge to transmit this dynamically-produced data to cloud centers, analyze it and provide useful results in real time. In order to provide safe and efficient transportation through this system we have to find a way to detect problems and provide real time traffic information to the users, control traffic lights to create green waves, etc. These scenarios require us to take into account their location as well.

2.4 Context

The Internet of Things comprises large numbers of smart devices at the network edge that may have to collaborate and interact with each other in real time. There is an ever increasing number of devices and they are usually handled by a cloud. The devices, which are connected to the Internet and the cloud, are becoming more intelligent and more heterogeneous. In addition, these devices are distributed over vast geographical areas and generate huge amounts of data. The IoT environment predominantly supports wireless access and mobility. In addition, the IoT often requires a multiplicity of service providers [Bonomi et al. 2014].

2.5 Problem

How do we provide compute, storage and networking services efficiently to applications running on IoT devices? The IoT is becoming increasingly popular and we have a growing number of heterogeneous smart devices connected to the Internet. The cloud usually provides support in these situations, however, it finds it increasingly difficult to handle these devices. The applications running on these devices can produce a huge amount of data and given the limited available bandwidth, it is difficult to transmit this data to the cloud, process it, and return actionable results in real-time. In addition to the volume of data and number of devices, wireless connectivity, mobility and geographical distribution make it difficult to get required services directly from the cloud. Cloud data centers are few in number and are sparsely located whereas devices are more widely distributed. The available bandwidth is limited and data transmission to the cloud data centers becomes a bottleneck.
Two major problems that IoT application faces when working with clouds are latency and bandwidth limitation. Moreover, to preserve the confidentiality and integrity of the data we also need to be able to apply policies for accessing and filtering the information.

The solution to this problem is guided by the following forces:

- *Large number of nodes*: Many devices are part of the networks and these nodes produce data which has to be collected and analyzed.
- *Latency*: Many applications run on devices at the edge of the network, which means they are at a distance from the few and far-located cloud data centers. Cloud computing supports centralized remote provisioning of resources which can introduce delay. Some of these applications can be latency sensitive and we want this latency to be as low as possible; otherwise, we could use cloud computing.
- *Mobility*: We want mobility support because many devices in the edge of networks are not stationary; for example, vehicles or people with smart devices. This should not affect the accessibility or efficiency of the system.

- *Location awareness*: End users may require applications that are aware of device location. For example, devices like smart traffic lights, vehicles and others. The information required by these applications can be specific to their geographical location. The cloud offers more global and centralized services and it can be difficult for cloud systems to maintain this level of location awareness with devices that are mobile and geographically distributed.
- *Heterogeneity*: Different devices that are part of the Internet of Things can have diverse components like routers, switches, access points, end user devices, follow different protocols, present different interfaces, and we have to support and manage them in a uniform and consistent way.
- *Transparency*: We do not want the users to be concerned about the storage, communication or computational limitations of their devices. Resource provisioning for the devices has to be transparent.
- *Big data analytics*: The volume of data produced by all edge devices is huge and it is impractical to transfer it to centralized data centers and analyze it there. When cloud provides all data analysis of IoT applications, it leads to inefficient bandwidth utilization and latency. There is a need to process a large part of the data being produced by the devices to provide results in real-time or almost real-time and to filter or remove noise from the data sent to the cloud.
- *Cloud support*: In order to provide low latency and location awareness we want a solution which is closer to edge devices but more permanent storage may be needed and long term computations might still have to be performed in cloud data centers. Interaction with the cloud has to be supported.
- *Scalability/Flexibility*: Resources and devices should be added dynamically to accommodate constant change.
- *Multi-tenancy*: Multiple applications have to be supported which requires resource sharing. We need multi-tenancy support in the computing platform which supports IoT applications. This may result in performance and security problems.
- *Multiplicity of providers*: IoT applications require support for a multiplicity of providers. IoT systems are distributed and it is likely that the system may extend beyond the boundaries of a single controlling authority, owner or provider. This requires the orchestration of consistent policies across multiple providers [Bonomi et al. 2012].
- *Security*: The devices may need to access shared databases in the cloud and we need to apply access control to this data, which also requires identity and authentication. Some devices may need even finer access control. We also need to control access to devices, in addition to access control to fog and cloud data.
- *Filtering*: The volume of data produced by the edge devices is huge. All of this data may not even be required by the cloud to provision required services; in fact, some data may be unnecessary or even potentially harmful if it is sent to the cloud. This can cause security issues and inefficient bandwidth consumption.

2.6 Solution

Introduce a platform to provide cloud computing-like services closer to the devices to be monitored or controlled. This is called Fog Computing. It provides computation, storage, and networking services between end user devices and cloud providers. Fog Computing can offer low latency, location awareness, efficient use of bandwidth and storage services. Data is processed locally for more immediate response. Aggregated data and other relevant information can be forwarded to the cloud for analysis. In addition, services like security, filtering, etc., can be provided by the Fog.

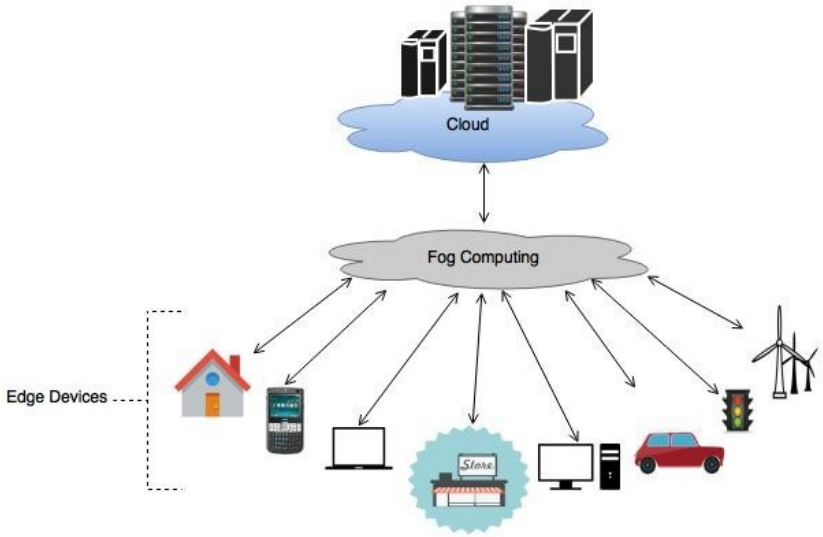

Fig. 2. Idea of Fog Computing

Figure 2 shows the idea of fog computing. It structures a way to handle devices, these devices can communicate with each other or directly with the cloud in some cases. Edge devices are smart devices which have Internet connectivity; for example, smart phones, tablets, laptops, traffic lights, vehicles, homes devices, etc. A Fog Computing platform provides low-latency virtualized services and is linked with the Cloud Computing infrastructure. There are many edge devices, a Fog Computing platform manages and controls these devices and the cloud providers are fewer in number. Size, storage capacity, processing capabilities, and latency increase as we move upward in the system.

The Fog acts as an intermediate layer between the Edge Devices and the Cloud. Edge devices request compute, storage and communication services from the Fog. The Fog provides local, low latency response to these requests and forwards relevant data for computationally intensive processing, long term analytics and persistent storage over to the cloud.

2.7 Structure

Figure 3 shows the class diagram for this pattern. The Fog is a collection of several distributed tiny clouds called Fog Nodes. They can be resource-rich servers, routers, access points, mobile devices etc. A Fog Node has resources including hardware (compute, networking and storage) capabilities. These nodes provide local real-time data analytics capabilities using an Analytics Engine. Applications can be hosted in the fog nodes using virtualization, Virtual Machine Monitor (VMM), Virtual Machine (VM) and/or Containers. A database stores both application data and necessary metadata for service orchestration. It also has information about hardware and software capabilities of nodes, information about the state of fog nodes and services, policies for security, filtering, and configuration. Fog computing uses a Reference Monitor to perform enforcement of Authorization policies. Depending on the information received, new policies can be added to the Policy Repository [Dsouza et al. 2014]. Data is transferred between fog nodes and the various components of the Fog. The Authenticator is used for providing access control to the system. In addition, services like filtering, aggregation of data, logging, etc., can be provided.

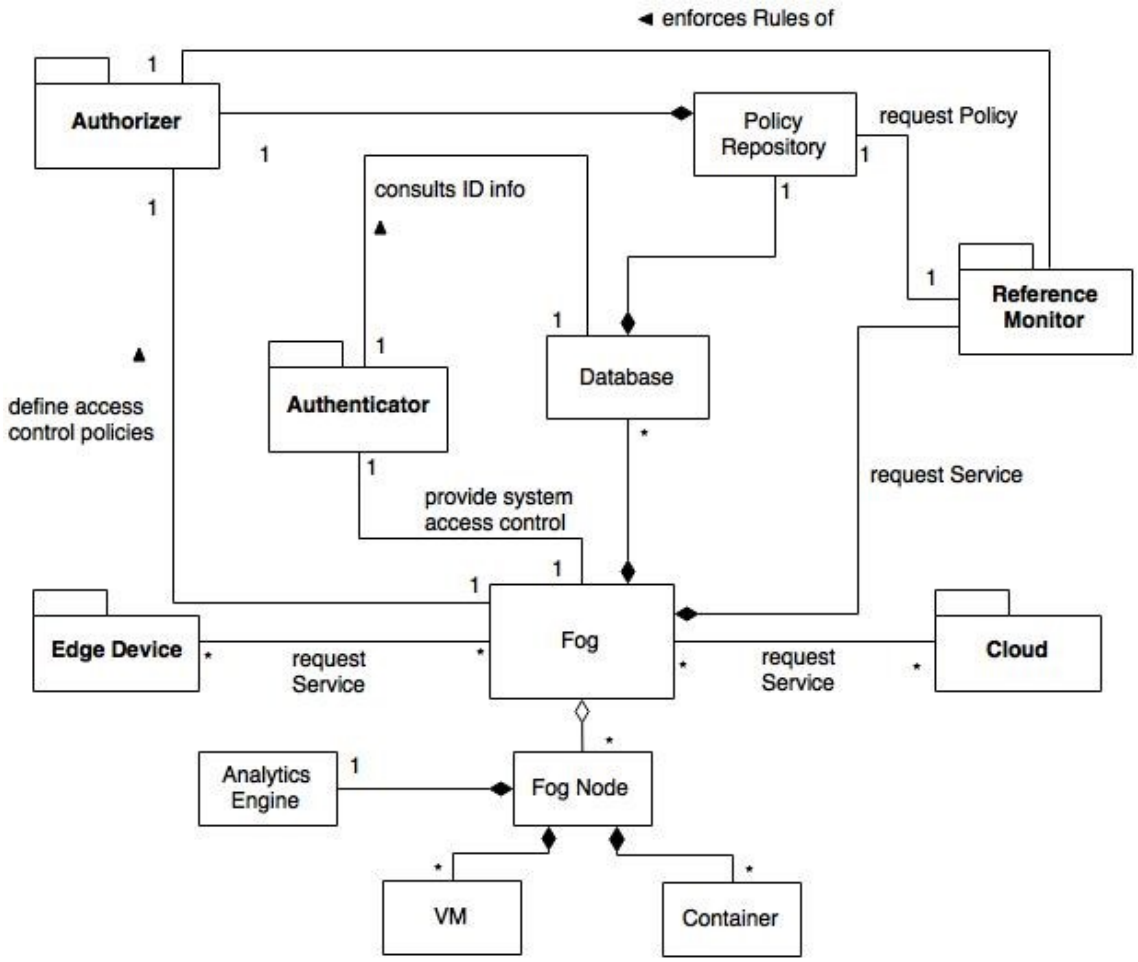

Fig. 3. Class diagram of the Fog Computing pattern

2.8 Implementation

- Fog computing requires multi-tenancy support for applications, from different software companies or developers, that need to execute without interfering with each other. So, the cloud platform in devices comprising the fog platform should support multi-tenancy for applications.
- Despite multi-tenancy, fog devices should be able to execute applications in isolation to prevent unwanted interference from other processes.
- Policies have to be defined to control service orchestration, filtering and for adding security.
- Decentralized management mechanisms need to be established to setup and configure a large number of devices in fog [Vaquero and Merino 2014].

Bonomi et al. [2014] mentions a local software agent on each fog node, called foglet, which monitors the state of the node and services. Foglets also perform lifecycle activities and orchestration. Also they mention a fog abstraction layer which provides uniform programmable interfaces for resource control and management. It provides generic APIs to monitor and manage heterogeneous fog devices.

Cisco Fog Computing with IOx can be used as an example of implementation of fog computing. Cisco IOx allows data analysis and command processing at edge of the network using “Cisco Data in Motion”,

"IOx middleware analytics" and "policy engine" [Cisco 2015]. Information is aggregated and filtered before sending it to the cloud. This conserves storage and bandwidth. Cisco IOx provides IoT application development and deployment with its SDK and middleware services. It enables application hosting through an infrastructure called "Application Enablement Platform". It also supports mobility between cloud and fog. In addition to this, Cisco provides application monitoring and management services for applications deployed in the Fog.

Each individual fog is assigned to a specific area in the same way that clouds have zones. This means that selecting a fog implies selecting a specific area, which makes location awareness an implicit part of fog computing. For example, if a router is used as a fog node it would have a particular location. In addition to this, Cisco IoT solutions offer Connected Mobile Experiences (CMX). These components triangulate device location using WiFi and track location via device's MAC address/IP address/Username if it has been registered.

Analytics engine in a product refers to the local analysis capability of fog nodes. However, analysis needs of applications can vary so they have added a service and configured it as part of application to meet particular needs. For example in case of Cisco, Data in Motion is being built into Cisco components and using a restful API it can be built into applications. It is configurable by these applications to analyze data, we can use it to find specific data, summarize it, process it, perform other application defined analysis, etc.

2.9 Known Uses

- Cisco IOx provides a fog computing unit which enables users to host OS and applications to control a variety of devices [Cisco 2015].
- Foghorn Systems provides products for industrial IoT applications. Foghorn technology platform includes an analytics engine for fog computing called Complex Event Processing (CEP) engine and a Domain Specific Language (DSL) to apply rules on the incoming sensor data streams. Aggregated data can be sent to the cloud using publishing functions or used for action at edge devices. It also provides a SDK for developing edge applications [Foghorn 2016].
- PrismTech's Vortex provides both fog and cloud computing products, providing real-time Device to Device (D2D) and Device to Cloud information sharing [Vortex 2015].
- ParaDrop is a new fog computing architecture on gateways, which uses home routers or WiFi access points) that can host the platform. This platform can be used by developers to design containers based on LXC [Willis et al. 2014].

In addition to these, other IoT solutions also have similarities to the solution discussed here. Amazon Web Services (AWS) launched its Amazon IoT cloud service for IoT [Amazon 2015], and IBM provides IoT analytics solutions [IBM 2016].

2.10 Consequences

This pattern presents the following advantages:

- *Large number of nodes:* Fog computing can offer densely distributed data collection points so it is easier and more efficient to handle this data locally.
- *Latency:* Fog platforms are closer to the end user device and therefore they offer low latency.
- *Mobility:* Fog computing can support mobile devices with densely and widely distributed fog nodes.
- *Location awareness:* Being closer to the network edge, Fog computing has location awareness. This enables applications to provide services better suited to user and device location.
- *Heterogeneity:* Fog can support heterogeneity of IoT devices by having an abstraction layer to provide uniform programmable interfaces for managing resources and controlling the devices [Bonomi et al. 2014].

- *Transparency*: End users are not burdened with resource limitation; for example, storage or computation limitations. The Fog computing platform will take care of "how and where to provide these services". Policy based orchestration functionality is provided by the fog platform in distributed fashion. If user needs more resources it will be provisioned by a fog node which has the required capabilities.
- *Big data analytics*: Fog computing allows data collection and analytics.
- *Cloud Support*: Immediate data processing needs can be fulfilled using fog computing; however, it does not substitute the cloud, rather complements it. Aggregated data is sent to cloud data centers for further processing
- *Scalability/Flexibility*: Fog computing supports dynamic addition or removal of devices.
- *Multi-tenancy*: Resources available on Fog devices can be shared between multiple applications.
- *Multiplicity of providers*: Fog devices can be owned by different providers which are coordinated from the fog platform [Bonomi et al. 2012].
- *Security*: Authentication and authorization services can be provided by the fog platform. Access to the user devices, fog devices and the cloud can be controlled. The fog platform can act as a reference monitor and performs policy-based control.
- *Filtering*: Information can be filtered before it is sent to the cloud to ensure effective utilization of bandwidth. Instead of transferring all data generated by the devices, only processed and aggregated data will be passed on to the cloud. Unnecessary or potentially harmful information can be removed before it is exchanged with the cloud. This not only protects the cloud but also adds privacy as far as origin of data is concerned.

Liabilities of the pattern include the following:

- Use of fog computing means device users will have less control over selection of fog service providers as compared to the cloud. A limited number of providers are involved in service provisioning in the case of clouds. However in the case of fogs, smaller fog servers will be larger in number and will be owned by different providers. Trust and security will most likely be more difficult to establish.
- Devices are now sharing resources in a more distributed environment. The confidentiality of the users can be a concern and mechanisms have to be adopted which provide authentication and authorization for ensuring user privacy. Authentication and authorization will be required at three levels: user devices, fog platform and cloud providers. Policies must be mapped between these levels. [Stojmenovic and Wen 2014] discusses security and privacy issues in fog computing.
- A multiplicity of providers may also lead to compliance problems, especially when health or financial data is involved.
- Fog computing should support multi-tenancy which can bring a potential security problem unless strong isolation is ensured between these applications.
- Heterogeneous mobile devices that are geographically distributed over a large area and are owned by different providers, could make management and configuration a problem as compared to the cloud where a centralized approach is used.

2.11 Example Resolved

The smart traffic control system is now using fog computing. Data from devices is collected and analyzed locally to ensure real-time response. The traffic system has improved in preventing accidents and users experienced fewer delays owing to better traffic light controls by the system. In order to monitor long term patterns in behavior of the system data is now aggregated and sent to the cloud. The system has become more efficient and also conserves bandwidth.

2.12 See Also (related patterns)

- Patterns for Cloud Computing Architecture [Fernandez 2013]: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).
- Cloud Security Reference Architecture [Fernandez et al. 2015] describes a security cloud reference architecture.
- Virtual Machine Operating System [Fernandez 2013]. Virtual machines are used to execute different operating systems with strong isolation between them.
- Software Container pattern, lightweight isolated execution environments for applications sharing same host OS [Syed 2015].
- Authenticator pattern [Fernandez 2013] allows verification of identity of subject intending to access the system.
- Authorization pattern [Fernandez 2013] describes who is authorized to access specific resources based on their identity, in a system.
- Reference Monitor [Fernandez 2013]. In a computational environment in which users or processes make requests for data or resources, this pattern enforces declared access restrictions when an active entity requests resources. It defines an abstract process that intercepts all requests for resources and checks them for compliance with authorizations

3. ACKNOWLEDGEMENTS

We thank our shepherd, Christopher Preschern, for his insightful comments that have significantly helped to improve this paper.

REFERENCES

Amazon. 2015. How the AWS IoT Platform Works. Retrieved February 15, 2016 from https://aws.amazon.com/iot/how-it-works/

Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. 2012. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing* (MCC '12). ACM, New York, NY, USA, 13-16. DOI=http://dx.doi.org/10.1145/2342509.2342513

Flavio Bonomi, Rodolfo Milito, Preethi Natarajan and Jiang Zhu. 2014. Fog Computing: A Platform for Internet of Things and Analytics. In *Big Data and Internet of Things: A Roadmap for Smart Environments*, Springer International Publishing Switzerland, 169–186. DOI:10.1007/978-3-319-05029-4_7

Jan Bosch. 2009. From Software Product Lines to Software Ecosystems. In *Proceedings of the 13th Int. Software Product Line Conference* (SPLC'09), ACM (August 2009) Carnegie Mellon University, Pittsburgh, PA, USA, 111-119.

F. Buschmann, R. Meunier, H. Rohnert, P. Sommerland, and M. Stal. 1996. *Pattern-Oriented Software Architecture. Volume 1: A System of Patterns.* Volume 1. Wiley, 1996.

Cisco. 2015. Cisco Fog Computing with IOx. Retrived August 8, 2015 from http://www.cisco.com/web/solutions/trends/iot/cisco-fog-computing-with-iox.pdf

Clinton Dsouza, Gail-Joon Ahn, and Marthony Taguinod. 2014. Policy-driven security management for fog computing: Preliminary framework and a case study. In *IEEE 15th International Conference on Information Reuse and Integration (IRI)*, IEEE (Aug. 2014), 16-23. DOI:10.1109/IRI.2014.7051866

Eduardo B. Fernandez, N. Yoshioka, H. Washizaki and M. H. Syed, "Modeling and security in cloud ecosystems", *Future Internet* 2016, *8*(2), 13; (Special Issue Security in Cloud Computing and Big Data), doi:10.3390/fi8020013

Eduardo B. Fernandez, N. Yoshioka and H. Washizaki. 2015. Patterns for Security and Privacy in Cloud Ecosystems. In *2nd International Workshop on Evolving Security and Privacy Requirements Engineering* (ESPRE 2015), IEEE (August 2015), 13-18. DOI:10.1109/ESPRE.2015.7330162

Eduardo B. Fernandez, Raul Monge, and Keiko Hashizume. 2015. Building a security reference architecture for cloud systems. *J. Requirements Engineering* (June 2015), 1–25. DOI:10.1007/s00766-014-0218-7

Eduardo B. Fernandez. 2013. *Security Patterns in Practice: Designing Secure Architectures Using Software patterns*. J. Wiley Sons, May 2013.

Foghorn. 2016. Foghorn Technology. Retrived October 20, 2016 from https://foghorn-systems.com/technology/

IBM. 2016. Watson Internet of Things: IoT in the cognitive era. Retrieved February 20, 2016 from http://www.ibm.com/internet-of-things/watson-iot.html

Ivan Stojmenovic and Sheng Wen. 2014. The Fog computing paradigm: Scenarios and security issues. In *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems* (FedCSIS 2014). IEEE, 1-8. DOI:10.15439/2014F503

M. H. Syed and E. B. Fernandez. 2015. The Software Container pattern. *22nd Conference on Pattern Languages of Programs*

(PLoP 2015). Pittsburgh, PA, Oct. 24-26, 2015.

Luis M. Vaquero, and Luis Rodero-Merino. 2014. Finding your way in the fog: Towards a Comprehensive Definition of Fog Computing. *ACM SIGCOMM Computer Communication Review*. 44, 5 (October 2014), 27-32. DOI:http://dx.doi.org/10.1145/2677046.2677052

Vortex. 2015. Fog Computing. Retrieved August 8, 2015 from http://www.prismtech.com/vortex/technologies/fog-computing

Dale Willis, Arkodeb Dasgupta, and Suman Banerjee. 2014. Paradrop: a multi-tenant platform for dynamically installed third party services on home gateways. In *Proceedings of the 9th ACM workshop on Mobility in the evolving internet architecture* (MobiArch '14). ACM, New York, NY, USA, 43-48. DOI:http://dx.doi.org/10.1145/2645892.2645901

Received January 2016; Revised March 2016; Accepted October 2016