International Journal of Security and Its Applications
Vol. 11, No. 12 (2017), pp.35-44
http://dx.doi.org/10.14257/ijsia.2017.11.12.04

Implementation of a Mobile Agent System to Detect DoS/DDoS Flooding Attacks in Cloud Computing

Abdelali Saidi[1], Elmehdi Bendriss[2] and Mohamed El Marraki[3]

[1,3]LRIT-CNRST URAC29,
Mohamed V University, Faculty of Sciences
4, Avenue Ibn Battuta, B.P. 1014 RP 10006 Rabat, Morocco
[2]UFR SI3M,
ENSIAS, BP. 6624
Al Irfane, Rabat 10112, Morocco
Abdelali.saidi@gmail.com, marraki@fsr.ac.ma and bendriss@gmail.com

Abstract

In previous works, we've proposed and described the functional aspects of a mobile agent system to help improve the early detection of the first signs of DoS/DDoS flooding attacks in Cloud environment. This system consists of some distributed components working altogether to help improve the safety of virtualized Cloud Computing environments. In this paper, we detail the algorithms of its main components; an analyzer module, an alert module, a security events module and a moving module. The proof of concept was simplified as possible as it can be to ensure the effectiveness of our concept. We have used the libpcap C library to gather Ethernet frames and process Ethernet, IP and ICMP headers. The simulations concern the behavior of our system when the virtual environment is suffering an ICMP flooding attack. The results show how well our system can be aware of the variation of the ICMP packets cadence.

Keywords*: Network Security; DoS/DDoS flooding attacks ; Mobile Agent ; Cloud Computing*

1. Introduction

Cloud availability is the main metric to evaluate a Cloud provider's quality of service, somehow particularly its level of security. Normally, a good Cloud services provider must not suffer more than 7h of unavailability per year.

The cloud providers use the latest technologies to deliver better options and flexibility. The virtualization is one of the most interesting technologies used in Cloud Computing environments. It helps to respond instantly to their client's demands.

Cloud services are accessible from the Internet; the most unpredictable network in the world. Akamai's state of the Internet reported some shocking statistics [3], a DDoS attack can reach an extent of 7 GB per second and last 21 hours in a row. This is a serious challenge for Cloud providers.

To detect DoS and DDoS in Cloud environment, there are some classic techniques in the literature that are very interesting:

1. Hop-count filtering: HCF is a filter dedicated to the classification of the traffic according to the number of jumps [14]. Initially, this filter was used to handle IP spoofing attacks, but since most DoS attack techniques send traffic with spoofed IP addresses, the filter can also be useful for detecting DoS attacks and DDoS.

2. Confidence based filtering: CBF is a technique that detects any deviation of traffic from its normal form [15]. It is also an improvement of the HCF technique that considers

Received (May 18, 2017), Review Result (October 25, 2017), Accepted (December 3, 2017)

ISSN: 1738-9976 IJSIA

different fields in the package. CBF is based on a certain correlation between these areas that can be considered after a period. This correlation builds a normal profile and the CBF filter tries to detect any deviation from it. To be sure, the server will wait for a packet, the client and the server must pre-share a key and divide the time into slots. At the beginning of each slot, the client must calculate the port number using an algorithm and pre-shared key.

3. Random port hoping: RPH is a technique that allows a server to change the port number when communicating with a legitimate client. First, this technique was used to hijack spies. Lee *et al* [16] used this technique to mitigate DoS and DDoS attacks.

4. IP trace back: IP Traceability is a technique that tracks stolen packets to determine their true origin.

DDoS attacks are usually used to deflect security devices from a main attack. We propose a system to lighten these devices work and take care of DDoS flooding attacks. This system is based on mobile agents. Each one of these mobile agents is responsible of a set of virtual machines (VM) and will analyses security events of these VMs alternately. Briefly, a mobile agent must run through some virtual machines and analyze every one of them. When it lands on a virtual machine, it must find and process some security events. Here, in every virtual machine we have a gathering module that filters arriving packets and save some header values when the type of the packet seems interesting. Nevertheless, the mobile agent may be processing a virtual machine while the attack may occur elsewhere. For this, anytime the mobile agent moves, it indicates the hosting machine on an information table, so, every virtual machine that is in distress. This affects the order of the lap taken between virtual machines.

The following of the paper is organized as follows: first, there is a section about the related works. Secondly, we explain in the third section the problematic treated by our approach and we give a general overview of the modules that construct our system. Thereafter, in the fourth section, we run through details concerning the role of every subsystem and how they may interact. The fifth section encloses the algorithms of the main modules. And in the sixth section, results of our simulations are showed and discussed.

2. Related Works

Multi-agent systems are suitable to build distributed intrusion detection system [5]. Akyazi and Uyar proposed four methods to detect intrusions, three of them are based on mobile agents [9]. Each method deploys Snort like a probe. The mobile agent contributions in these works are the reduction of the network load thanks to bringing the analyzing near to the source of the attacks. Huang *et al* [6] published a demonstration on how multi-agent systems are precise and rapid. Agents can cooperate with security and administration devices. The concept of agent mobility opened another dimension for scientific research in intrusion detection. Venkateshwaran and al propose some algorithms to ensure the security of the interaction of mobile agents in the Cloud environment [10]. In [7], O'Malley *et al* show that for the same work, mobile agent systems are 5 times faster than multi-agent systems. The slowness is caused by communications and its maintenance. Subsequently, several studies have adopted this mobility within their dIDS. Zamani and al propose a mobile agent system based on danger theory [11] to build a system that's able to face DoS/DDoS attacks. Kannadiga and Zulkernine [8] achieved a comparison between a mobile agent based dIDS and a central IDS (Snort). Mobile security agents were also raised in the latter work and the proposed solution is to authenticate mobile agents and encrypt data with a Java applet. Duraipandian and Palanisamy propose a distributed architecture based on mobile agents to detect different attacks [12]. The mobile agents will be deployed in different important nodes of the Internet to be the nearest to the source of the attack. Demir and al propose an

enhancement of the DoS/DDoS detection by demonstrating the importance of the locations in where mobile agents can be deployed [13].

3. The Mobile Agent System

3.1. The Objective of the Proposed System

Nowadays, the cloud computing environment is usually based on virtualization technologies. The benefits are numerous [9] which makes it easy for Cloud administrators to offer new services, lower the downtime caused by maintenance operations and all other benefits of using virtualization in a Datacenter. Using dedicated physical machines to host services is no more a best practice in today's information systems usages. This is why in our architecture we based the proposed solution on Virtual Machines (VM) environment. Like we said in the introduction, the mobile agent must run through some virtual machines. These latter may be deployed on the same physical server or not. For the sake of an easier implementation of the proof of concept, we will limit the discussion to VMs on a same physical server.

In a real environment, VMs are not solicited the same way, so it is obvious that analyzing a VM hosting a public web server will likely take more time than another VM hosting a service used internally or by a limited set of clients. Figure 1 represents a minimal and classic Cloud provider infrastructure.

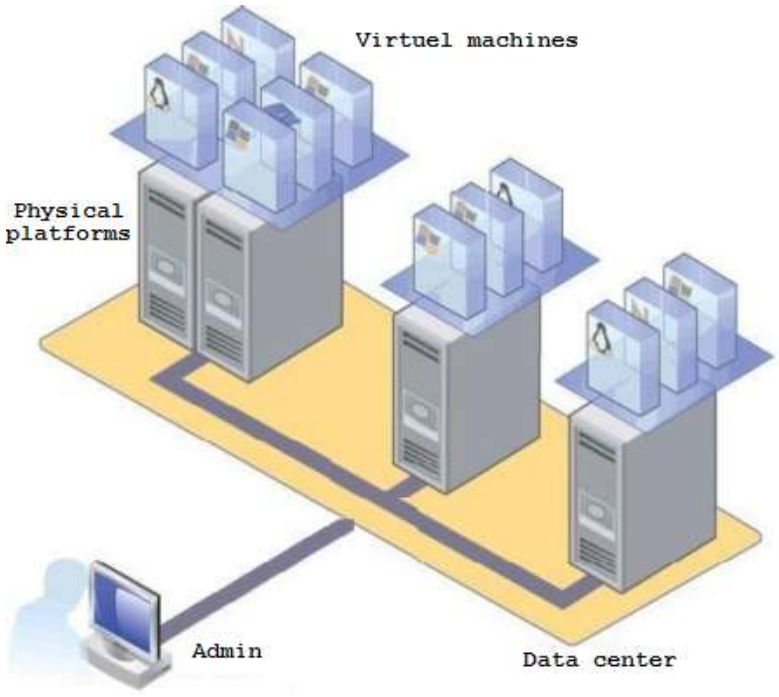

Figure 1. A Basic Cloud Provider Infrastructure

Hence, VMs are grouped in different sets depending on how they must be checked. Every set will be handled by one mobile agent. So, we can notice that the number of VMs in a set affects the quality of service of its mobile agent. The more VMs a mobile agent must handle, the more time it will take to comeback on the same VM. In other words, the more VMs are in a set the less frequently the agent will check every VM. Now that we gave a general idea of our system, we will go into more details in the rest of our paper.

3.2. The Operating Mode of the Proposed System

The proposed system is made of a set of subsystems. Each one has a precise role and can be established whether on the VMs, the mobile agent or the manager device. Figure 2 describes the possible interactions between the subsystems.

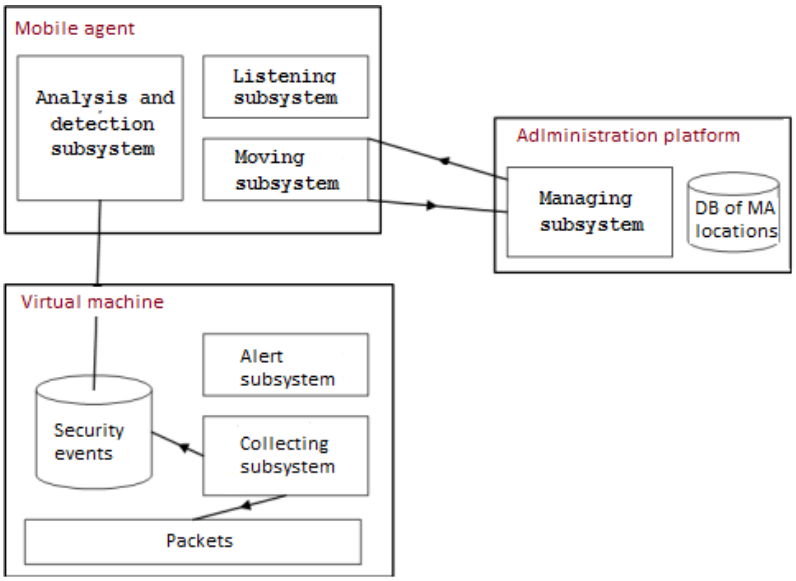

Figure 2. Mutual Interaction of the Components of Our System

* Security event collector: is deployed on every VM and is responsible of the gathering of information related to protocols ICMP, TCP and UDP. For ICMP, we collect only the ping request. For TCP, we collect every SYN request that hasn't been acknowledged by the client. For UDP, we collect every datagram;

* Alert subsystem: is also deployed on every VM, it helps to contact the mobile agent if the VM that hosts this subsystem is in a critical state. This subsystem can't guess the location of the mobile agent, so it contacts the manager subsystem;

* The manager subsystem: is deployed on a manager device. It is responsible of how the mobile agent moves from a VM to another one and keeps trace of its location. On its execution, the mobile agent doesn't know all the VMs in its set; it only contacts the manager subsystem to get back the next VM ID it has to visit. This latter is the one with the most ancient date of visit;

* Moving subsystem: is a component of the mobile agent. It helps to contact the manager subsystem to get back the next VM to visit when the mobile agent is done with the current one. Nevertheless, this is not the only reason for what the mobile agent will move, it can be because of a VM that is on critical state;

* Listening subsystem: is also a component of the mobile agent. It helps it to get aware of an alert coming from a VM who's in a critical state;

* Analysis subsystem: is also a component of the mobile agent. It is responsible of analyzing security events that are collected on each VM. For our proof of concept, an alert will be triggered based on a threshold.

4. General System Functional

Three parts are principle for our system: the mobile agent, the virtual machine and a scheduling part. In every part, we place at least one module of our system. Figure 2 gives a global idea of this parts components and how the subsystems interact generally.

4.1. The Probe Subsystem

This subsystem is the responsible module of filtering received packets and to gather the events that can interest the mobile agent when this one arrives. The probe subsystem is implemented in every monitored VM. It turns the Network Access Controller of the VM in promiscuous mode, sniffs and filters packets and save ICMP, UDP and TCP headers values in the event security database. This latter must be created in every VM.

4.2. The Scheduling Subsystem

To monitor some VMs, we can't make the same mobile agent to move between a VM hosting a football web site to a VM hosting an e-commerce web site. It is obvious that the degree of criticism isn't the same. So, some constraints have to be taken into considerations before regrouping some VMs in the same monitoring group: The analyze duration for every VM and the number of the VMs of a group. These two constraints define the quality of service of the mobile agent. Also, the scheduling subsystem manages the assignation of the VMs to the mobile agents. It records the IDs of the VMs belonging to the monitoring space of every mobile agent with the last moment of visit. Thanks to this, when a mobile agent finish analyzing a VM, this subsystem sends the ID of the next VM to analyze to the mobile agent.

4.3. The Analyze and Detection Subsystem

The Analysis and Detection Subsystem is the smart part of our system. It allows the mobile agent to analyze the security events of a virtual machine and make a decision in case of abnormal activity. Overall, there are two main approaches to intrusion detection: the anomaly approach and the signature approach. The last two aim to distinguish between legal and illegal traffic. There are many methods for detection, such as: statistical methods, flexible computer methods, knowledge-based methods, data mining methods, and machine learning methods [8], which is common to all these methods. apply a thresholding mechanism. belongs to an attack.

To validate our model, we also apply a thresholding mechanism. The analysis consists in calculating the number of packets received in a time interval and the decision to compare this number to a threshold. If the number of packets exceeds the threshold, the mobile agent will send an alert.

4.4. The Listening Subsystem

The listening subsystem allows virtual machines (which are not being scanned by the mobile agent) to contact the mobile agent if necessary. A VM that is abnormally used sends a request to the scheduling subsystem. The latter is the one who manages the order of the VMs. It takes into account the alert, modifies the order of the virtual machines and sends the identifier of the next virtual machine. In the case where the mobile agent analyzes a VM that is in a critical state, rather than moving, the mobile agent clones itself and sends the clone to the VM which is also in an abnormal state, and thus right now.

4.5. The Moving Subsystem

The moving subsystem allows the mobile agent to communicate with the scheduling subsystem and retrieve the ID of the next virtual machine to be scanned. The mobile agent moves from one virtual machine to another when it completes the scan of a virtual machine or receives an alert from a virtual machine. In the first case, the mobile agent contacts the moving subsystem to return the ID of the next virtual machine. In the second case, the VM sends an alert to the moving subsystem. The latter contacts the mobile agent to immediately switch to this virtual machine.

5. The Subsystems Algorithms

5.1. The Manager Subsystem

This subsystem acts on three main sides: the allocation algorithm that help to choose the set where a new VM will be placed, En ending algorithm that a help a mobile agent to recuperate the identity of the next VM to visit and the receiving alert algorithm that help a mobile agent to receive alerts from VMs that are in critical states.

1) Allocation algorithm:
   * Estimation of the VM treatment duration
   * Service quality estimation of the mobile agent on groups with
   * VMs similar to the new one
   * VM assignment to one of these groups
2) Endings algorithms
   * Reception of end of treatment from the mobile agent
   * Querying the database to retrieve the identifier of the VM with the oldest date of visit
   * Sending a move message to the mobile agent
   * Store current VM identity
   * Save the new date of visit
3) Algorithm receiving alerts*
   * Reception of a warning from a VM
   * Reception of the identifier of the mobile agent responsible for VM
   * Viewing the current location of the mobile agent
   * Sending alert to the mobile agent

5.2. Security Event Collector

This subsystem collects security events from three types of packets ICMP, TCP and UDP. It has to record information stored in the header fields of each data unit on a database. This latter has three tables; each one concerns a protocol type.

* Security event tables:
   * ICMP: store the source IP address, type, code and the date of reception;
   * TCP: store the source IP address, the destination port and the date of reception;
   * UDP: store the source IP address, the destination port and the date of reception.
   * The algorithm:
   * Opening the network interface
   * Applying the expression filter
   * Test on the protocol fields of each captured packet
   * Connecting to the database
   * Storing information

5.3. Analyze Subsystem

This subsystem sends an alert each time a VM receives a number of ICMP/TCP/UDP packets that exceeds a threshold. This latter depends on how the VM is sought by its clients. It's up to the administrator to properly well choose this value. The algorithm is:

   * Connecting to the database
   * Counting ICMP/TCP/UDP packets time intervals
   * Comparing each result with the threshold
   * If (ICMP / TCP / UDP) traffic exceeds the threshold, an alert notification will be sent.

6. Results of the Proof of Concept

We didn't implement yet the whole proposed system, but we have done a first test by implementing its two main subsystems; namely, the event collector and the analyze subsystems. These tests were done with using four VMs; one is used as the target, a second is used as the attacker, the others and the host machine are used as legitimate clients.

6.1. Test of the Event Collector Subsystems

Figure 3 shows how this subsystem responds to ICMP ping requests of different extents. On the left, the VM received only legitimate ping requests. On the right, we had proceeded to launch an ICMP flooding attack using hping3 [4]. In either ways, the subsystem was able to detect and store security event on the database.

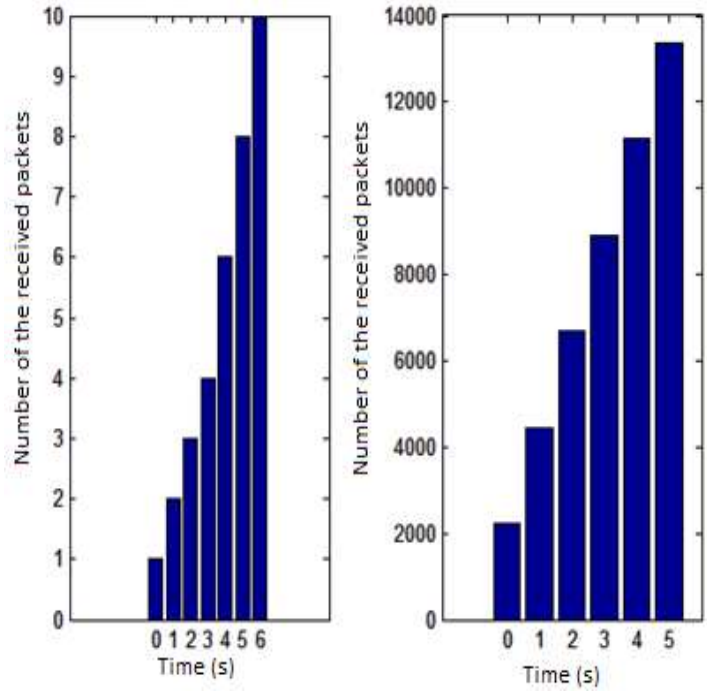

Figure 3. Reception of ICMP Packets

6.2. Test of the Analyze Subsystem

This subsystem searches the information stored on the security event database and tries to detect the ICMP flooding. For this purpose, it calculates the duration of every 10 ping requests. When this duration is very low, the subsystem proceeds the same way but with every 100 ping requests. If it is also very low, it proceeds with 10000 ping requests and so on. Figure 4 shows the results of this mechanism.

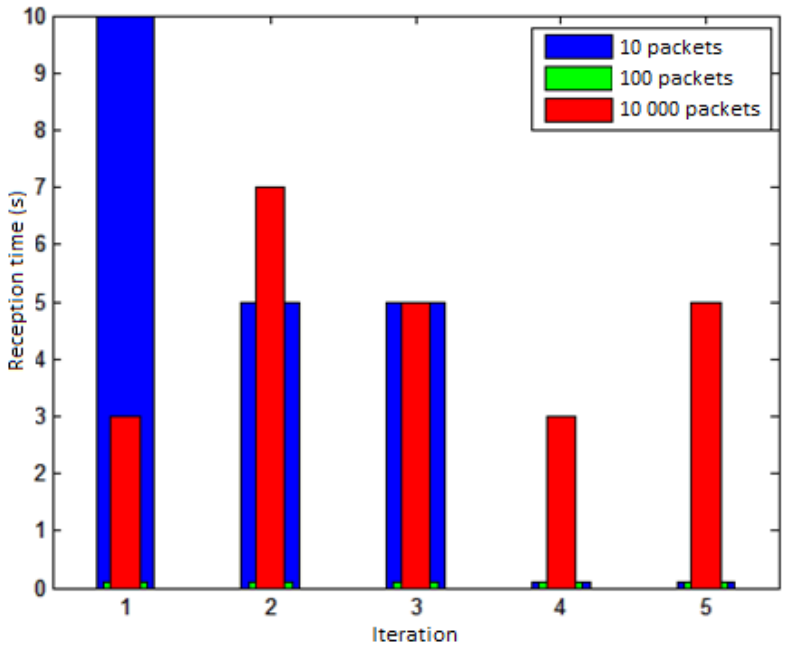

Figure 4. Analysis of Received ICMP Packets

The number of packets to calculate or the threshold that helps to detect the attack depends on how the target (*i.e.* The VM targeted by the ping requests) is solicited. In our test, the target is on a LAN with four other machines. So it is legitimate that we would then suspect and notify for an attack with the reception of the first 100 ping requests. Besides, we can see in the graph that for 100 packets the duration is too much low which is not normal in normal conditions. When the system moved to 1000 ping requests, we can clearly see that the duration is too low to not consider it as an attack.

7. Conclusion

The objective of our work is to increase the availability of Cloud services by enhancing the detection of DDoS flooding attacks. Our proposed system tries to take advantage of mobile agents and their ability to work in a distributed and collaborative way to keep an eye on Virtual Machine permanently. The simulations ensure the good response of our system when a deviation of the threshold value occurs. Future works will aim to first complete the implementation of our system. We have to analyze security events concerning SYN and UDP flooding attacks. We expect to integrate our system with an existing intrusion detection system. Like we said before, DoS and DDoS attacks are usually launched to deflect targets attentions from an organized and principle attacks. We want to see how our system can help the main IDS node to detect DoS/DDoS flooding attacks and relieve it so that it takes care of other more intelligent attacks.

References

[1] I. M. Hegazy, T. Al-Arif, Z. T. Faye and H. M. Fa- heem, “A multiagent based system for intrusion detection”, IEEE Potentials, vol. 22, no. 4, **(2003)**, pp. 28–31.
[2] hping3, “send (almost) arbitrary tcp/ip packets to network hosts”, tech. rep.
[3] W. Huang, Y. An, W. Du and H. M. Faheem, “A multi-agent based distributed intrusion detection system”, in In 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE 2010), **(2010)**, pp. 3141–3143.
[4] P. Kannadiga and M. Zulkernine Didma, “a dis- tributed intrusion detection system using mobile agents”, in In Sixth International Conference on Soft- ware Engineering, Artificial Intelligence,

Networking and Parallel/ Distributed Computing and First ACIS International Workshop onSelf-Assembling Wireless Network, **(2005)**, pp. 238–245.
[5] S. A. OMalley, A. L. Self, and S. A. Deloach, "com- paring performance of static versus mobile multiagent systems", in In National Aerospace and Electronics Conference (NAECON 2000) , **(2000)**, pp. 282–289.
[6] A. Saidi, E. Bendriss, A. Kartit and M. El Marraki, "A mobile agent system to enhance dos and ddos detection in cloud computing", European Journal of Sci- entific Research, vol. 131, no. 2, **(2015)**, pp. 209–214.
[7] A. Saidi, E. Bendriss, A. Kartit and M. El Marraki, "The functional of a mobile agent system to enhance dos and ddos detection in cloud", International Journal of Applied Engineering Research, vol. 11, no. 6, **(2016)**, pp. 4615–4617.
[8] Akamais state of the internet. "Q2 2015 report secu- rity", tech. rep., **(2015)**.
[9] H. Salimi, M. Najafzadeh and M. Sharifi, "Advantages, Challenges and Optimizations of Virtual Machine Scheduling in Cloud Computing Environments", International Journal of Computer Theory and Engineering, vol. 4, no. 2, **(2012)**, pp. 189–193.
[10] U. Akyazi and A.S.E. Uyar, "Distributed intrusion detection using mobile agents against ddos attacks", In Computer and Information Sciences, 2008. ISCIS '08. 23rd International Symposium on, **(2008)**, pp. 1–6.
[11] K. Venkateshwaran, A. Malviya, U. Dikshit and S. Venkatesan. "Security Framework for Agent-Based Cloud Computing", International Journal of Interactive Multimedia and Artificial Intelligence, vol. 3, no. 3, **(2015)**.
[12] M. Zamani, M. Movahedi, M. Ebadzadeh and H. Pedram, "A ddos-awareids model based on danger theory and mobile agents", In Computational Intelligence and Security, 2009. CIS '09.International Conference on, vol. 1, **(2009)**, pp. 516–520.
[13] M. Duraipandian and C. Palanisamy, "An intelligent agent based defense architecture for ddos attacks", In Electronics and Communication Systems (ICECS), 2014 International Conference on, **(2014)**, pp. 1–7.
[14] O. Demir, B. Khan, G. Ben Brahim and A. Al-Fuqaha, "Optimizing agent placement for flow reconstruction of ddos attacks", In Wireless Communications and Mobile ComputingC onference (IWCMC), 2013 9th International, **(2013)**, pp. 83–89.
[15] C. Jin, H. Wang and K. G. Shin, "Hop-count filtering: An effective defense against spoofed ddos traffic", In Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS '03, New York, NY, USA, ACM, **(2003)**, pp. 30–41.
[16] W. Dou, Q. Chen and J. Chen, "A confidence-based filtering method for ddos attack defense in cloud environment", Future Gener. Comput. Syst., vol. 29, no. 7, **(2013)**, pp. 1838–1850.
[17] H.C.J. Lee and V.L.L. Thing, "Port hopping for resilient networks", In Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th, vol. 5, **(2004)**, pp. 3291–3295.

Authors

Abdelali Saidi, he is a PhD in Computer science and telecommunication at Mohammed V University in Morocco since 2017. He has a Master degree in Systems and Networks from Ibn Tofail University. He teaches in the area of computer science (Linux, IP Networks, and Information Security) since 2012. His main interest is Cloud Computing Security researches.

Elmehdi Bendriss, he received his PhD degree in Networks Security from ENSIAS, Mohamed V University in 2014. He also holds MSC in IT from the same University and an engineering diploma from INPT since 2002. He's now working on systems and networks Security and especially in Cloud computing. He's been teaching in the area of Computer Science (Systems and Networks administration, Information Security, Virtualization) since 2003.

Mohamed El Marraki, he received the Doctorate and the Doctorate of the State degrees in algebra and number theory, respectively, from the Bordeaux University, France in 1991, and the Mohammed V-Agdal University, Rabat, Morocco, in 1996; he also received the Doctorate in "dessin d'enfant theory" from the Bordeaux University, France in 2001. He joined Mohammed V University, Rabat, Morocco, in 1996, first as an associate professor and full Professor since 2000, where he is teaching. Over 19 years, he developed teaching and research activities covering various topics of Mathematics.