MULTI-AGENT BASED INTRUSION DETECTION SYSTEM FOR CLOUD COMPUTING

J. O. Yogo
Department of Computer Science, Nairobi University, Nairobi, Kenya
E-mails: jimmy.yogo@gmail.com

Abstract
Cloud Computing has come to represent a significant shift in Information Technology. Companies are excited by the opportunities to reduce capital costs, invest on core competencies and as well by the agility offered by the rapid on-demand provisioning of computing resources. However, security concerns have slowed down the uptake and trust has been identified as the key barrier to the uptake by most small to medium sized enterprises. This paper details a multi agent based framework for intrusion detection in the cloud computing environment that provides a mechanism for auditability and accountability. The design and development of the framework is based on PASSI (Process for Agent Societies Specification and Implementation) methodology. The proposed design consists of a virtual environment used to simulate the cloud computing environment, client and server side agent (java based) systems setup in the virtual instances. The outcome of the experiments carried out reveal that monitoring of all user operations on files predefined for monitoring within the virtual instances was successful. Email and SMS notifications were sent in real time to either clients or administrators for any operations detected. The framework depicts aspects of security assurance that assists in the efforts to promote trust between providers and consumers of cloud computing hence increase the uptake of the technology. The potential beneficiaries of the implementation of the framework are both service providers and consumers of cloud computing.

Key words: Agents, multi-agent systems, cloud computing, virtual machine, infrastructure as a service, intrusion detection system

1.0 Introduction

1.1. Background

With the promising innovations, cloud computing has come to represent a significant shift in Information Technology. Companies are both excited and nervous at the prospects of cloud computing. They are excited by the opportunities to reduce capital costs, invest on core competencies and as well by the agility offered by the on-demand provisioning of computing services, (Fujitsu Research Institute, 2010). In Gartners' report, "Gartner Top End User Predictions for 2010, coping with the New Balance of Power ", 2010, it is observed that by the year 2012, 20% of businesses will own no IT assets; while the need for computing hardware will not go away, the actual ownership of it will shift to cloud computing providers.

Most modern computer systems incorporate some form of long-term storage, usually in the form of files stored in a file system. These files typically contain all of the long-lived data in the system, including user data and applications, and system executable and databases. As such, the file system is one of the usual targets of an attack (Gene *et al*., 1994). Motives for altering system files are many. Intruders could modify system databases and programs to allow future entry and as well system logs could be removed to cover their tracks or discourage future detection. Modification or destruction of user files also compromise aspects of the security policy. As such, the security administrator and the owner of data need to closely monitor the integrity of the file system contents.

According to a report by Cloud Security Alliance (2010), "Top 10 threats to Cloud computing v10, 2010" and a report by Brodkin (2008) on "Seven cloud-computing security risks", Abuse and Nefarious Use of Cloud Computing, Insecure Interfaces and APIs, Malicious Insiders, Shared Technology Issues, Data Loss or Leakage, Account or Service Hijacking and Unknown Risk Profile are amongst the top security risks in a cloud computing environment. A good percentage of these risks narrow down to lack of proper mechanisms of auditability and accountability.

A recent survey by Fujitsu Research Institute (2010), on "Personal data in the cloud: A global survey of consumer attitudes, 2010)," revealed that 88% of potential cloud consumers surveyed are worried about who has access to their data within the cloud, and would like to have more awareness of what "goes on" in the cloud's backend

physical servers. Such surveys have not only identified trust as the key barrier to cloud computing uptake, but also enhanced the urgency for researchers to quickly address key obstacles to trust.

Despite auditability and accountability being a crucial component of improving trust and confidence, current prominent providers e.g. Amazon EC2/ S3, Microsoft Azure and others are still not providing full transparency or capabilities for tracking and auditing of file access, (Ryan *et al.,* 2011).

Concerns about the risks of cloud computing remain a serious issue that needs to be addressed. From a system design perspective, the notion of trust can be increased through reducing risks when using the cloud. While risks can be greatly mitigated via privacy protection and security measures such as encryption, they are not enough, particularly as full encryption of data in the cloud at present is not a practical solution. There is a need to complement such preventative controls with equally important detective controls that promote transparency, governance and accountability of the service providers, (Ryan *et al*., 2011).

Hence, in this paper, we identify accountability and auditability as urgent research areas for the promotion of trust in a cloud computing environment. We propose a multi agent based framework for intrusion detection in a cloud computing environment (with our focus on the *IaaS* model) which will fully addresses risks related to accountability and auditability in a cloud environment. We embrace the PASSI (Process for Agent Societies Specification and Implementation) methodology in modeling the proposed agent based system as it offers step by step requirements to code guidelines and the much needed iterativeness to ensure user requirements are fully captured, (Massimo and Luca, 2003).

1.2. Problem Statement

The ability to provide multi-tenant cloud services at the infrastructure, platform, or software level is often underpinned by the ability to provide some form of virtualization to create economic scale. However, the use of the vitalization technologies brings additional security concerns and current prominent providers are still not providing full transparency or capabilities for auditing and auditability, (Ryan *et al,.* 2011), such that:

(i) One can never tell which and when file deletions and modifications within a subscribers' virtual space are conducted and by who, as there is no effective mechanism for monitoring authorizations and changes to subscribers' data within the cloud.

(ii) One can never get to know the exact changes done to files within a subscribers' virtual space as there is no effective mechanism for tracking changes (whether authorized or not authorized) done to subscribers' data within the cloud.

(iii) According to a report by Fujitsu Research Institute (2010), "Personal data in the cloud: A global survey of consumer attitudes, 2010," 88% of potential subscribers surveyed are worried about who has access to their data within the cloud and would like more awareness of what 'goes on'. The survey identifies trust as a key barrier to cloud computing uptake.

1.3. Significance of the Study

The project will prove the effectiveness of agents systems in the area of intrusion detection monitoring in a cloud computing environment; hence will:

(i) Provide a mechanism for auditability and accountability in the cloud computing environment.

(ii) Provide real time notifications to the cloud administrators or subscribers in terms of either short messages or email in case of modifications or deletions.

(iii) Detect and reduce a percentage of security breaches in a cloud computing environment hence enhance the level of trust between the provider and the client.

1.4. Scope and Delimitations of Study

The study only focuses on Infrastructure as a Service (IaaS) model of the cloud, and to be specific the storage aspect of it. The study does not focus on Platform as a Service (PaaS) and Software as a Service (SaaS) models of the cloud.

1.5. The Proposed Design

The proposed system (Figure 1) is based on multi agents and offers a solution to auditability and accountability to the IaaS model of cloud computing. Multi agents systems can operate asynchronously and in parallel, and this can result in an increased overall speed (provided that the overhead of necessary coordination does not outweigh this gain). They are scalable as they can be adapted to an increased problem size by adding new agents, and this does not necessarily affect the operations of the other agents.

The conceptual model of the proposed system comprises:

(i) A VMware environment with a number of instances enough to demonstrate the multi-tenant nature of cloud computing.

(ii) Database Server which hosts the information about changes or deletions on files defined to be monitored for any intrusions.

(iii) SMS Server used for sending SMS notifications to the cloud computing subscriber or administrators for the set triggers.

(iv) Email Server used for sending email notifications to the cloud computing subscriber or administrators for the set triggers.

(v) Reporting Server used for generating reports for consumption by the cloud computing administrators.

(vi) Client agents - responsible for monitoring file creations, modifications or deletions in the subscribers' virtual space, collecting the changes detected and pushing to the processor agent in real time.

(vii) A processor agent - responsible for parsing, classification and saving the changes received from the client agents to the database, it then as well relays the messages to the communicator agent.

(viii) A communicator agent –responsible for sending email or short message notifications to relevant predefines parties.

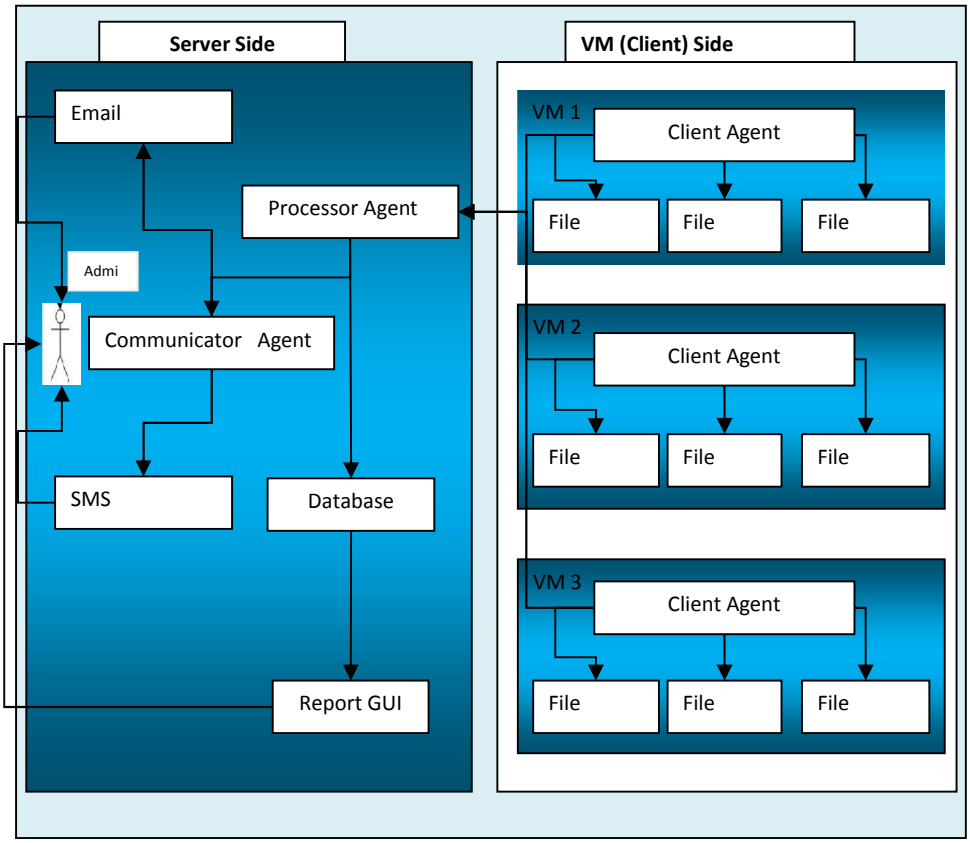


Figure 1: Conceptual model of the proposed system

1.0 Materials and Methods

1.1 Methodology

Process for Agent Societies Specification and Implementation (PASSI) methodology, (Massimo, 2005) which is an Agent Oriented methodology was used for the design and development of the multi-agent societies and integrating design models. The design process is composed of five models as illustrated below, (Massimo and Luca, 2003):

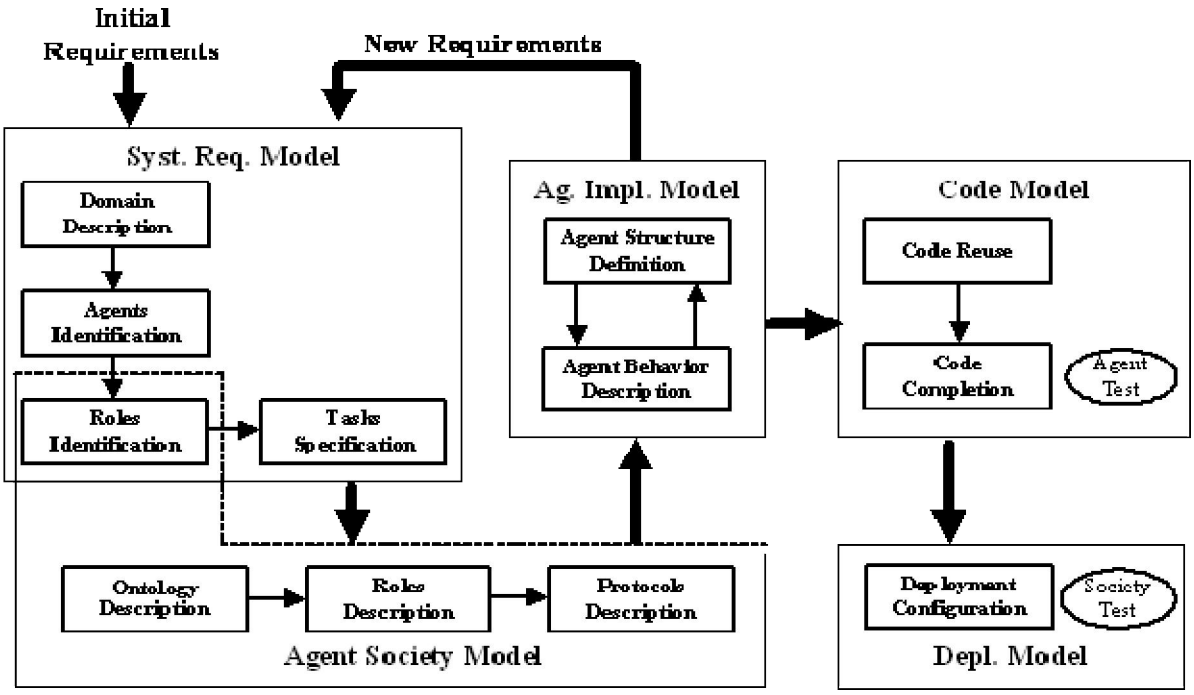


Figure 2: The PASSI methodology (Massimo and Luca, 2003)

1.2 System Analysis and Design

The PASSI methodology was used to model the design (Figure 1 to 10). This was achieved through the construction of five models obtained by performing twelve sequential and iterative activities as per the methodology.

1.3 Design Implementation

To implement the proposed design; JADE was used as the development framework of the modelled design, JSMS java API for SMS notification, SQL for database, .net and C# for Graphical User Interface (Figure 6).

1.4 Systems Testing

Testing was categorized into 2, i.e. verification and validation testing. For verification or unit testing, the behavior of each agent was tested with regards to the original requirements. Each executable java class identified was tested. Verification testing was conducted by the developer of the system. The following list describes the features that were tested:

Table 1: Test tasks-verification testing

Test Number	Description
AT-001	Monitoring of directories or files for any user activity (create, modify or delete)
AT-002	Running backup of the files to be monitored
AT-003	Comparing files being monitored against the backed up ones
AT-004	Parsing, classification and sending of changes
AT-005	Saving of changes to database
AT-006	Send email notification to the administrator
AT-007	Send sms notification to the administrator

For validation testing or society testing several related classes were tested together to ensure successful execution and compliance with the requirements after integration hence ensure that all functional requirements are met. The following list describes the functionalities that the system was tested against:

Table 2: Test tasks -validation testing

Test Number	Description of Functionality
ST-001	Detection of user activity (create operation) on monitored files and reporting on web GUI. [Report on the time of operation, user id, ip address of client VM instance, file changed and the exact change made]
ST-002	Detection of user activity (modify operation) on monitored files and reporting on web GUI. [Report on the time of operation, user id, ip address of client VM instance, file changed and the exact change made]
ST-003	Detection of user activity (delete operation) on monitored files and reporting on web GUI. [Report on the time of operation, user id, ip address of client VM instance, file changed and the exact change made]
ST-004	Detection of user activity (modify operation) on monitored files and sending a notification through email. [Report on file changed and the exact change made]
ST-005	Detection of user activity (delete operation) on monitored files and sending a notification through SMS. [Report on file deleted]

2.0 Verification and Validation Results

2.1 Sample Summary of Verification Results

The output presented below is part of the outcome of verification tests conducted to confirm the functionality of the various classes that make up the whole solution.

Test Number	Class Description	Expected Outcome	Actual Outcome	Status
AT-001	Watchdir class	All files in the directory predefined for monitoring should be monitored for any create, modify or delete operation.	Only (all) directories or files predefined for monitoring were monitored for create, modify or delete user operations to confirm the correct functionality of the *Watchdir* class.	Functional
AT-002	Backup class	All files in the directory predefined for monitoring should be backed up in the path specified.	Only (all) directories or files predefined for monitoring were backed up according to the preset schedule to confirm the correct functionality of the *Backup* class.	Functional
AT-003	FileCompare class	All modification changes in all the predefined directories to be monitored should be captured.	All or any modifications on monitored files were detected and any changes made picked out to confirm the correct functionality of the *FileCompare* class.	Functional
AT-004	MsgProcessor class	All file operations; create, modify and delete on any file in the monitored directory should be captured and reported on.	All file operations were detected and reported on accordingly as either a create, delete or modify operations to confirm the correct functionality of the *MsgProcessor* class.	Functional

2.2 Sample Summary of Validation Result

The output presented below are a sample summary of the outcome of some of the test cases from users who participated in the validation of the solution.

Test Number	Description of Test	Expected Outcome	Users Response	Users View
ST-001	Monitoring of create operations	All details about create operations on files in the monitored directory should be captured and saved to the database and report available on the reporting interface.	Create operations by user 1, 2,3 and 4 on monitored files from the different VM instances were reported on web GUI. Details recorded include; the time of operation, user id, ip address of client VM instance and file created.	Functional
ST-002	Monitoring of modify operations	All details about modify operations on files in the monitored directory should be captured and saved to the database and report available on the reporting interface.	Modify operations by user 1, 2, 3 and 4 on monitored files from the different VM instances were reported on web GUI. Details recorded include; the time of operation, user id, ip address of client VM instance, file modified and the exact changes made.	Functional
ST-003	Monitoring of delete operations	All details about delete operations on files in the monitored directory should be captured and saved to the database and report available on the reporting interface.	Delete operations by user 1, 2, 3 and 4 on monitored files from the different VM instances were reported on web GUI. Details recorded include; the time of operation, user id, ip address of client VM instance and file deleted.	Functional

2.3 Detailed Outcome

Figures 11 to 13 for sample detailed outcome.

3.0 Discussions of Results

From the results of the experiments carried out, 3 observations were made:
When a user executed a create operation on a file in a directory set to be monitored in a given instance of VM client, details of that operation such us the user id, time the operation was conducted, ip address from where the operation was executed, type of change and the file name of the affected file was recorded to the database. The same was the case for create operations executed in different VM instances with different user ids. These in turn can be monitored on the graphical user interface.

When a user executed a modify operation a file in a directory set to be monitored in a given instance of VM client, details of that operation such us the user id, time the operation was conducted, ip address from where the operation was executed, type of change, the exact change made and the file name of the affected file was recorded to the database. In addition to that, an email notification was also sent out to the predefined recipient showing the file changed and the changes made. The same was the case for modify operations executed in different VM instances with different user ids. These in turn can be monitored on the graphical user interface.

When a user executed a delete operation a file in a directory set to be monitored in a given instance of VM client, details of that operation such us the user id, time the operation was conducted, ip address from where the operation was executed and the file name of the affected file was recorded to the database. In addition to that, an SMS notification was also sent out to the predefined recipient showing the file deleted. The same was

the case for delete operations executed in different VM instances with different user ids. These in turn can be monitored on the graphical user interface. Email and SMS notifications are sent in real time hence administrators can react in time to arrest a case of an intrusion.

4.0 Conclusion

After successfully developing, implementing and testing the prototype; it is evident that the multi agent based solution for file centric intrusion detection in a cloud computing environment is effective for the intended purpose hence I would strongly recommend its' adoption by cloud computing providers. This in turn could be extended as a value add to consumers or be sold as 'Security as a Service' hence detect and reduce security breaches around auditability and accountability in the cloud computing environment to enhance the level of trust between the providers and the consumers, hence promote the uptake of cloud computing services.

Acknowledgments

Endless thanks goes to the Lord Almighty for all the blessings he has showered onto me, which has enabled me to write this last note in my research work. During the period of my research, I have been blessed by the Almighty with some extraordinary people who have spun a web of support around me. First and Foremost, I would like to thank my supervisor, Dr. Opiyo Omulo for his ideas, insight, guidance and support throughout my project work at the University of Nairobi. Secondly, I would like to thank my MSc project committee members who gave useful input while I was working on my research work and the entire University of Nairobi, School of Computing and Informatics fraternity for providing me with a rich and constructive environment to pursue my research interests.

References

Cloud Security Alliance (2010). “Top 10 threats to Cloud Computing v10.” Available: https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf

Thomas, J. B. (2009). Gartner: Server Virtualization; One path that Leads to Cloud Computing. Gartner Core Research Note G00171730.

Gartner (2010). “Gartner: Top End User Predictions for 2010: Coping with the New Balance of Power.” Available: http://www.widality.com/gartner.pdf

Brodkin, J. (2008). “Gartner: Seven cloud-computing security risks.” Available: http://www.infoworld.com/d/security-central/gartner-seven-cloudcomputingsecurity-risks-853?page=0,1

Fujitsu Research Institute (2010). “Personal data in the cloud: A global survey of consumer attitudes.” Available: http://www.fujitsu.com/downloads/SOL/fai/reports/fujitsu_personal-data-in-the-cloud.pdf

Fujitsu Research Institute (2010). “Personal data in the cloud: The Importance of Trust.” Available: http://www.fujitsu.com/downloads/WWW2/news/publications/FSL-0016_A4_TrustReport_online-final.pdf

Wang G., DeMara Ronald F., Rocke Adam J. “A Centralized Control and Dynamic Dispatch Architecture for File Integrity.” Available: http://www.iiisci.org/journal/CV$/sci/pdfs/P101422.pdf

Massimo Cossentino, Luca Sabatucci (2003). “Modeling Notation Source PASSI.”

Heiser, J., Nicolett, M. (2008). “Gartner: Assessing the Security Risks of Cloud Computing.” Available: http://cloud.ctrls.in/files/assessing-the-security-risks.pdf

Cyber Security Operations Center Initial Guidance, Department of defense, Australia (2011). “Cloud Computing Security Considerations.” Available: http://www.dsd.gov.au/publications/Cloud_Computing_Security_Considerations.pdf .

Ryan, K. L., Ko, Peter J., Miranda, M. and Siani, P. (2011). “Trust Cloud: A Framework for Accountability and Trust in Cloud Computing.” Cloud and Security Lab Hewlett-Packard Laboratories.

Gene, H. K., Eugene, H. S. (1994). “Experiences with Tripwire: Using Integrity Checkers for Intrusion Detection”.

Mell, P. and Timothy, G. (2011). Cloud Computing Security Considerations.

Gerhard, W. (1999). “A Modern Approach to Distributed Modern Approach to Artificial Intelligence.” Massachusetts Institute of Technology.

Nikos, V. (2003). “A Concise Introduction to Multiagent Systems and Distributed AI.” Informatics Institute, Universiteit van Amsterdam.

Massimo, C. (2005). “From Requirements to Code with the PASSI Methodology.”

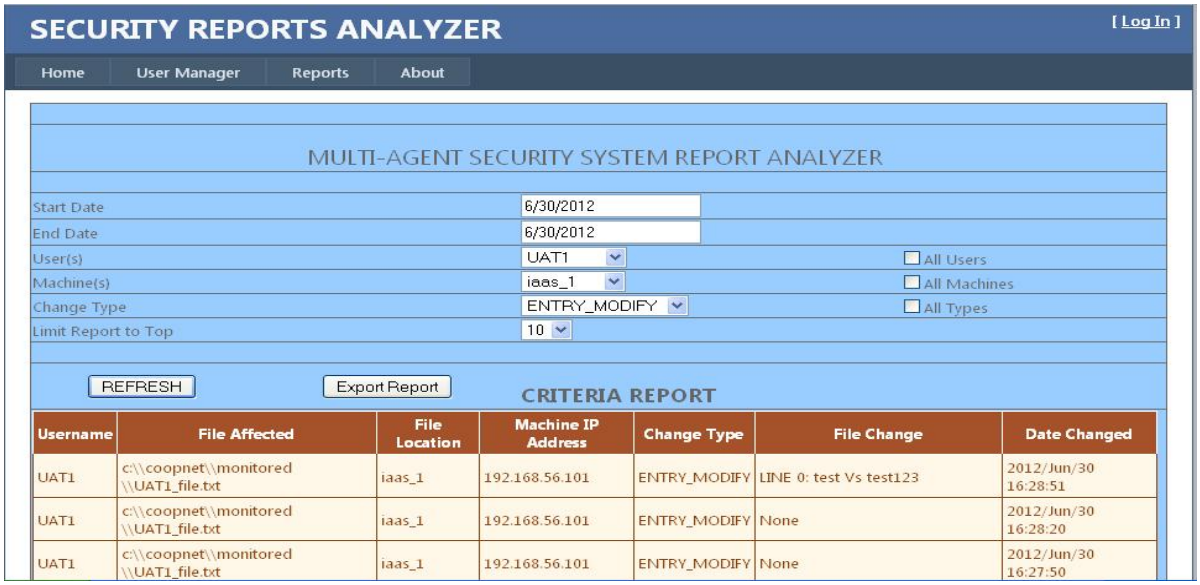


Figure 3: Main page –graphical user interface

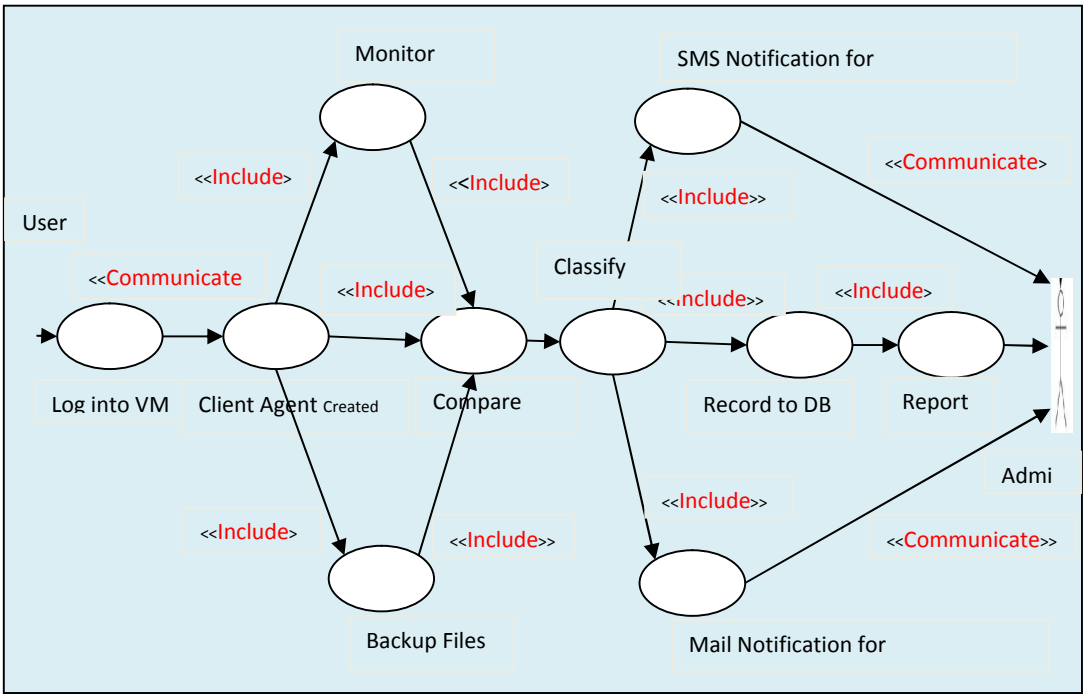


Figure 4: Domain requirement description diagram

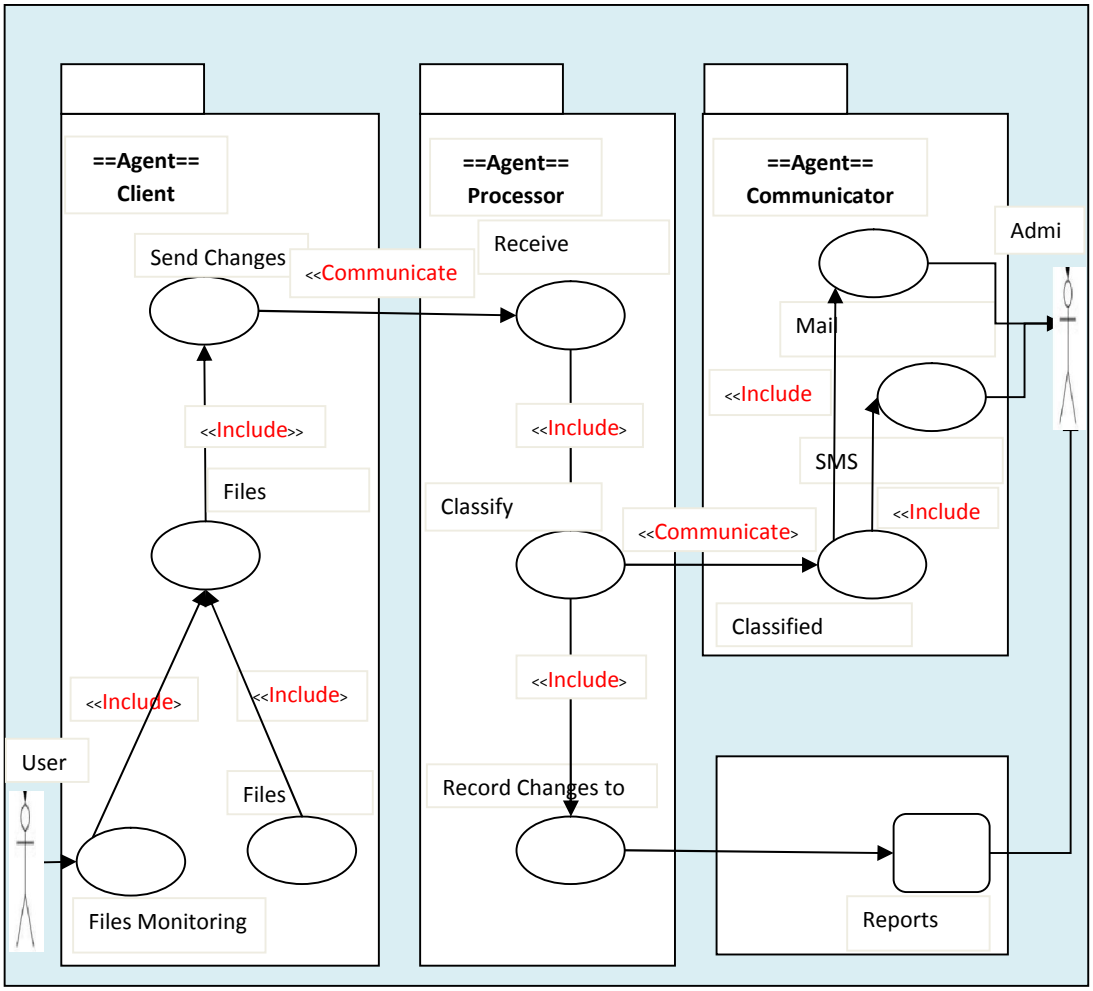


Figure 5: Agent identification diagram

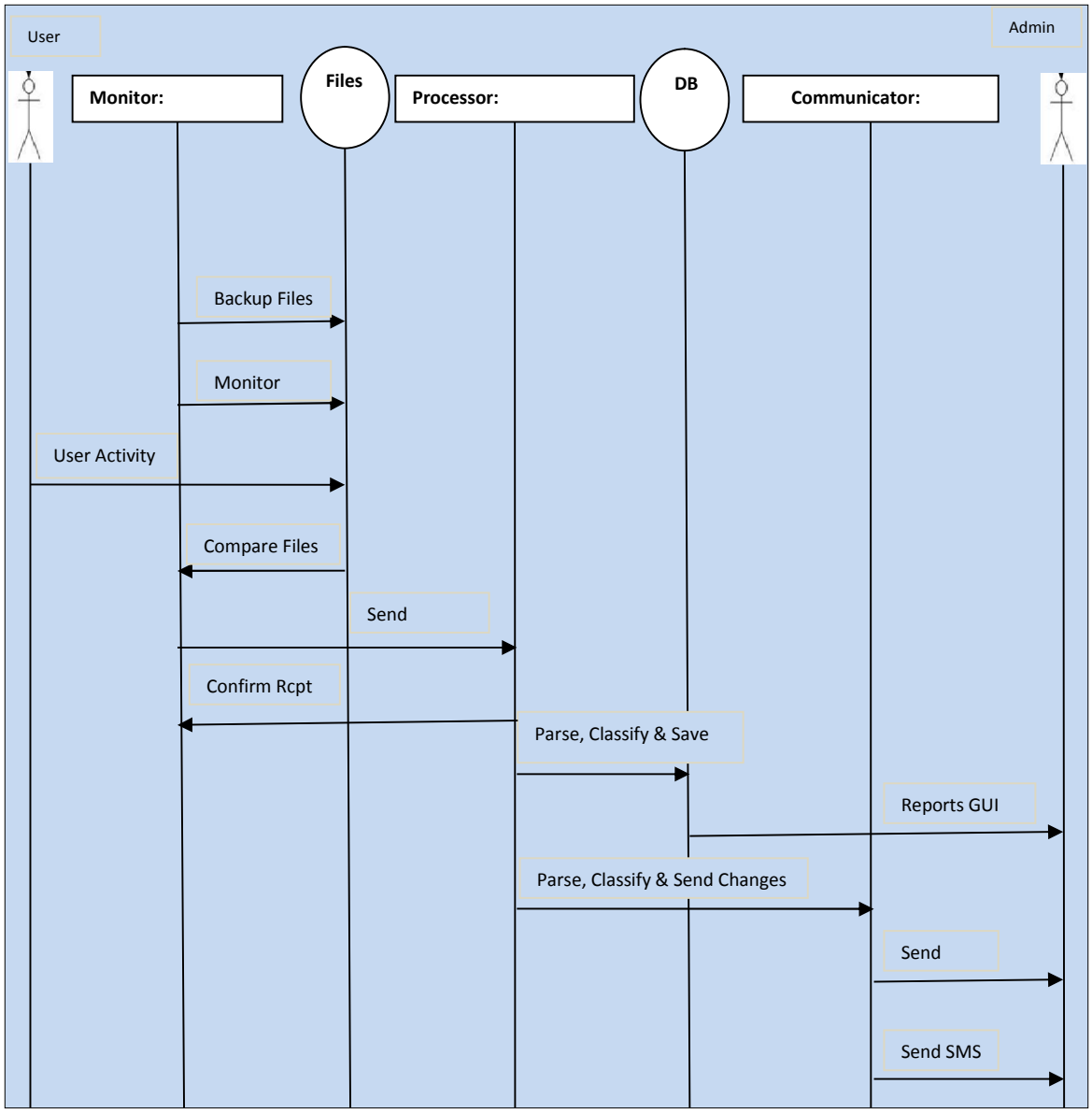


Figure 6: Roles identification diagram

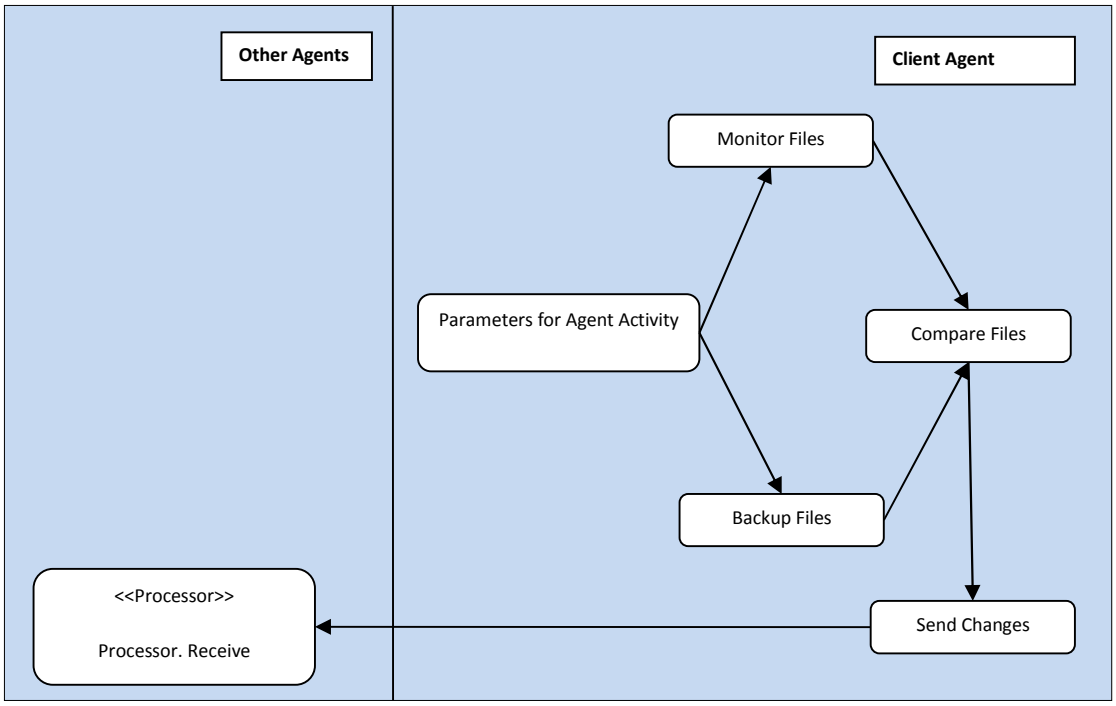


Figure 7: Tasks specification diagram- client agent

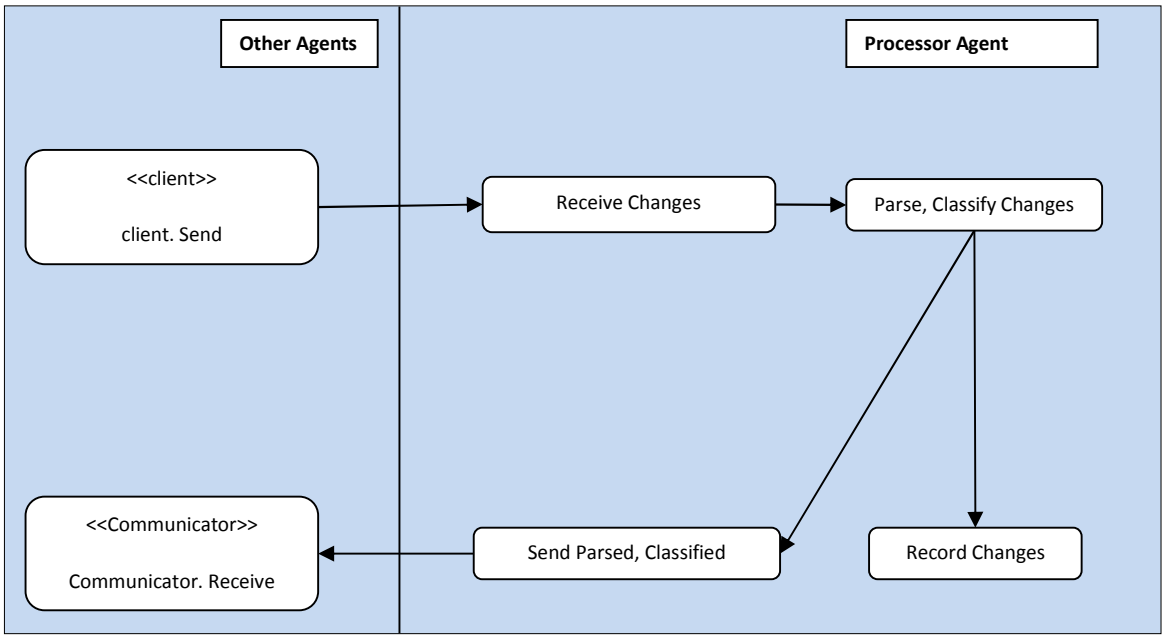


Figure 8: Tasks specification- process agent

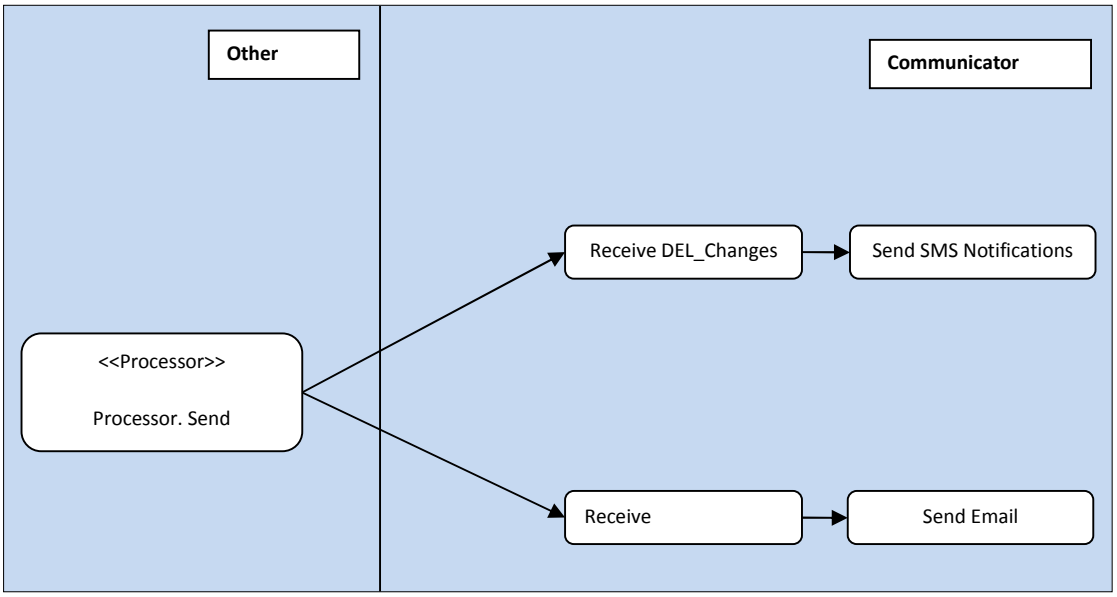


Figure 9: Tasks specification- communicator agent

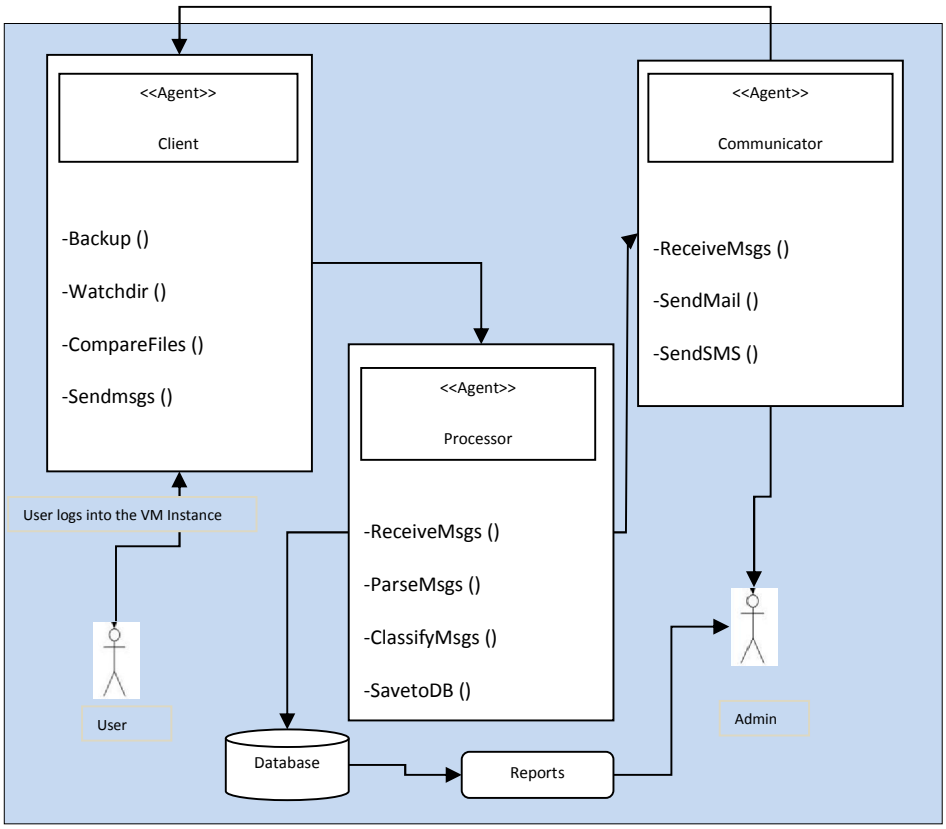
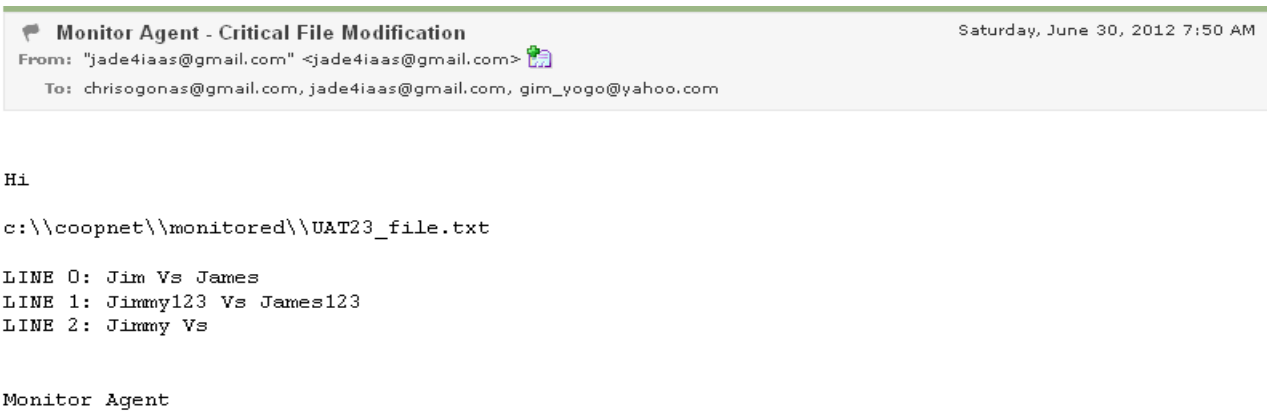


Figure 10: Multi agent structure diagram

UAT22	c:\\coopnet\\monitored\\UAT22_file.txt	iaas_2	192.168.56.102	ENTRY_DELETE	None	2012/Jun/30 11:54:33
UAT22	c:\\coopnet\\monitored\\UAT22_file.txt	iaas_2	192.168.56.102	ENTRY_MODIFY	LINE 0: test Vs test123	2012/Jun/30 11:53:31
UAT22	c:\\coopnet\\monitored\\UAT22_file.txt	iaas_2	192.168.56.102	ENTRY_MODIFY	LINE 0: Vs test	2012/Jun/30 11:52:43
UAT22	c:\\coopnet\\monitored\\UAT22_file.txt	iaas_2	192.168.56.102	ENTRY_MODIFY	None	2012/Jun/30 11:51:47
UAT22	c:\\coopnet\\monitored\\UAT22_file.txt	iaas_2	192.168.56.102	ENTRY_CREATE	None	2012/Jun/30 11:51:47
UAT22	c:\\coopnet\\monitored\\New Text Document.txt	iaas_2	192.168.56.102	ENTRY_DELETE	None	2012/Jun/30 11:51:47
UAT22	c:\\coopnet\\monitored\\New Text Document.txt	iaas_2	192.168.56.102	ENTRY_CREATE	None	2012/Jun/30 11:51:41
UAT21	c:\\coopnet\\monitored\\UAT21_file.txt	iaas_2	192.168.56.102	ENTRY_MODIFY	LINE 0: test Vs test123	2012/Jun/30 11:47:43

Figure 11: Sample validation test logs



Monitor Agent - Critical File Modification Saturday, June 30, 2012 7:50 AM

From: "jade4iaas@gmail.com" <jade4iaas@gmail.com>

To: chrisogonas@gmail.com, jade4iaas@gmail.com, gim_yogo@yahoo.com

Hi

c:\\coopnet\\monitored\\UAT23_file.txt

LINE 0: Jim Vs James
LINE 1: Jimmy123 Vs James123
LINE 2: Jimmy Vs

Monitor Agent

Figure 12: Sample validation test email

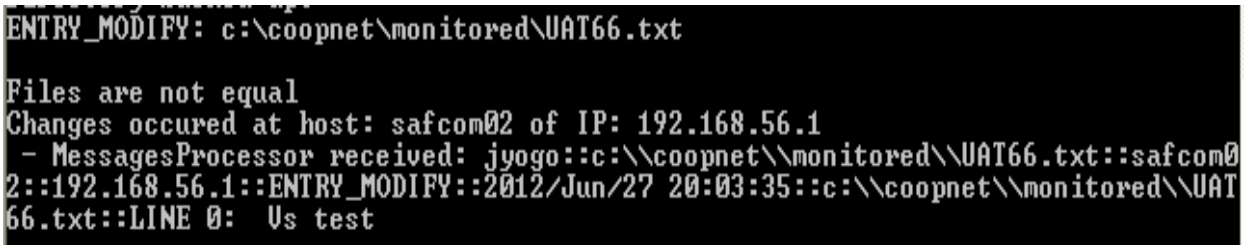


Figure 13: Sample verification test outcome (Watchdir class)