Aggressive and Intelligent Self-defensive Network Towards a New Generation of Semi-autonomous Networks

Ali Elouafiq[1,2],Ayoub Khobalatte[1,2], Wassim Benhallam[2],
Omar Iraqi[2], Tajje-eddineRachidi[2]

[1]Blue Sky Information Security
Email: {ceo, cto}@blueskysec.com

[2]Al Akhawayn University in Ifrane (AUI)
School of Science and Engineering
Computer Science Departments
Ifrane, Morocco
Email: {a.elouafiq, a.khobalatte, w.benhallam}@aui.ma

***Abstract*—Aggressive and Intelligent Self-defensive Network (AISEN) is an open-source distributed solution that aims at deploying a semi-autonomous network, which enables internal attack deception through misguidance and illusion. In fact, instead of simply preventing or stopping the attack as do traditional Intrusion Prevention Systems (IPS), AISEN drives attackers to attack decoy machines, which clone victim machines by mimicking their personalities (e.g. OS, services running). On top of that, AISEN uses rogue machines that clone idle production machines, which are able to detect human-aware zero-day attacks not seen by IPS. The solution uses real-time dynamic high-interaction honeypot generation, and a novel rerouting schema that is both router and network architecture independent, along with a robust troubleshooting algorithm for sophisticated attacks. Information captured and data gathered from these decoy machines will give CERTs/CISRTs and forensic experts critical data relevant to the sophistication of the attack, vulnerabilities targeted, and some means of preventing it in the future. This project reviewed former designs and similar studies addressing the same issues and emphasizes the added value of this open source solution in terms of flexibility, ease of use and upgrade, deployment, and customization.**

***Index Terms*—Network, IDS, IPS, Firewall, Incident Response, Antivirus, zero-day, advanced persistent threats, internal attacks, honeypot, state-of-the-art, enterprise-class**

I. Introduction

Up to this day, many innovative and useful systems have been created offering functionalities that help securing networks. Examples of these would intrusion detection systems (IDS), intrusion prevention systems (IPS), security information event managers (SIEMs), and honeypots. The goal of this project is to use these already existing technologies in accordance with each other and, as a result, provide an automatic self-defense mechanism that can be easily integrated in current networks.

Aggressive and Intelligent Self-Defensive network (AISEN) is an all-in-one solution that enables the different components in a given network to communicate with each other about the network's current state. In other words, it is a solution that intuitively reuses the already implemented security mechanisms available today in the market. The alarms generated by current SIEMs when an attack is detected are sent to a management server. These standardized alarm events should contain information about the type of the undergoing attack and the identities of the attacker and the victim, among other useful information. The management server then notifies the user agent installed at the level of the victim's machine, and asks it to redirect traffic to an appropriate centralized NAT Box. The choice of the NAT Box depends on the VLAN of the attacker and victim and on its availability. The NAT Box in turns forwards to the appropriate virtual honeypot for data capture and analysis. This insures the victim machine is not harmed, while keeping the attacker unaware of the rerouting mechanism and providing the security administrator with an overview of what is going on in the network.

In this paper, we will explain the most important features of AISEN, its components as well as it work flow.

II. System Architecture

A. System Overview

Aggressive and intelligent self-defensive network (AISEN) is a distributed security system that integrates different communicating components to ensure the network will benefit from live security monitoring, real time risk mitigation, and network obfuscation and deception. Essentially, AISEN relies on vendor neutral security information event management systems, third generation honeypot technology, and a distributed software architecture comprising intermediate network nodes, endpoint software agents, and a central management server. Following is a non-exhaustive list of features provided by AISEN:

Attack detection & prevention.

AISEN relies heavily on vendor-independent SIEM technology available today. Most of these SIEMs are equipped with intrusion detection and prevention systems, which monitor the network, generate standardized events, and communicate them in real time to AISEN's management server (discussed later). Essentially, SIEMs represent a fundamental starting point for all features provided by AISEN.

Network Deception and Illusion.

While securing the network and protecting the production environment, one critical feature is to prevent attacks. AISEN accomplishes this by providing obfuscation and diversion by means of an open source software emulating different network stations: Honeyd.

Attack rerouting.

One of the fundamental features of AISEN is the ability to dynamically reroute attacks targeting our production environment to automatically generated honeypots in the Virtual Environment. This feature is truly a breakthrough in the field of network security, as it constitutes a paradigm shift from traditional approaches to static honeypot-based network security.

Automatic honeypot generation.

This feature of AISEN entails the automatizing the process of launching, resuming, shutting down, or putting on hold all virtual machines in the Virtual Environment. This is possible because most of the virtual servers (i.e. machines that host honeypots) offer APIs to handle remote virtual machine handling. These honeypots can be either low or high interaction.

- *Real time discreet honeypot interaction upgrade.*

AISEN deals with attacks first using low interaction honeypots since they can be quickly set up and configured. However, an escalation towards high interaction honeypots in crucial to provide real system behavior, data capture, and, as a result, avoid compromising the rerouting process.

- *Virtual Environment Invisibility.*

Unlike traditional honeypot-powered security, AISEN does not allow any entity in the network to discover or communicate with honeypots directly. Instead, all incoming and outgoing traffic to/from the honeypots pass through a process called the NatBox. Furthermore, the Virtual Environment (the set of honeynets with honeywall) is put in a separate VLAN that only the NatBox can access. This ensures the Virtual Environment invisibility and therefore helps in the obfuscation of AISEN's internal structure.

Data Capture, control, and analysis.

As aforementioned, AISEN relies on honeypots for data capture and analysis. However, implementing this functionality at the level of each single honeypot would be redundant and time consuming. This is the reason why all traffic to honeypots pass through a layer-2 machine named honeywall. This machine keeps track of all traffic for future data analysis.

Attackers profiling.

Keeping track of attacks in the network insures attackers are identified and their behaviors kept track of for a predetermined period of time. This feature is called attackers profiling. However, AISEN only helps identifying these attackers; how to deal with these attackers is left to the network administrator.

Zero-day threat discovery.

The system will allow us to discover zero-day threats such as new malwares, rootkits, locale exploits when the attacker will fall into the trap of the machine that mimics the real one. Moreover, if an attacker's behavior is suspicious, we have a chance to capture a zero day attack that will not be detected by the IPS/IDS or the SIEM technology involved [8].

Strike back (Future work).

The strike back is a still debated concept in the realm of information security. AISEN includes an optional module that can be used should the strike back be deployed in future networks. The Strike Back mechanisms are built upon the idea of putting fake sensitive information and vulnerabilities in the network to attract possible attackers. If such attacker is identified, the strike back module of AISEN refers to its built-in extensible knowledge base that contains different scripts to confuse the attacker or, in extreme cases, shut down their system.

Distributed troubleshooting.

Perhaps the most important feature of AISEN is the distribution of different modules in the network. This ensures the integrity of the network if one of these modules shuts down. Refer to rerouting workflow section.

The use of IDS and IPS in modern network is essential. Some SIEMs even include these in a package and, consequently, offer the administrator a total view of what's happening, live, in the network. AISEN is built on this fact as t assumes the existence of an entity which job is to monitor the network all the time.

B. System Components

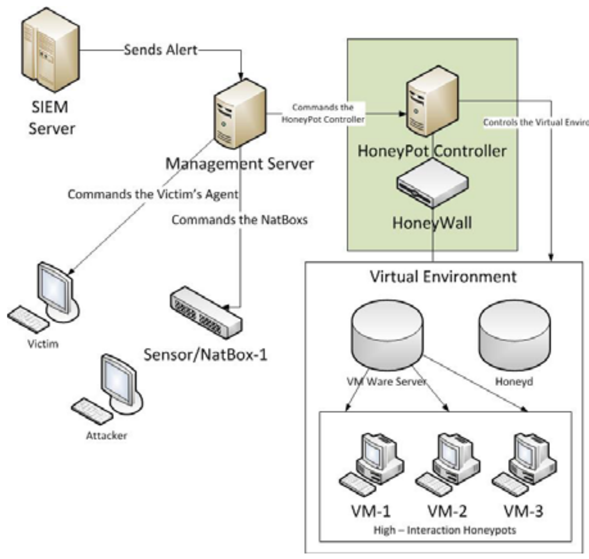

Fig. 1. AISEN's components

SIEM with intrusion detection and event generation.

The initial information gathering concerns both the attacker and the target. However, this only refers to basic characteristics such as collecting suspicious traffic and auxiliary information that describes the traffic, the events, and the incidents. Clearly, the information collected at this step is the result of a thorough analysis of the malicious activities detected by the IPS/IDS. Identifying and locating both the attacker and the victim is the intended outcome. In fact, the information collected about the victim host is tremendously important and will be used in the next step to create an environment that mimics its services and fingerprint [10].

Management Server.

The Management server is the central brain of AISEN; it receives alerts from the related SIEM, processes the alert, and chooses the appropriate decision based on the degree of the alert and its master configuration. The Management Server logs all the alerts received along with the respective decisions taken, is responsible for managing the workflow, and detecting failures and errors in NatBoxes and User Agents. The management server should be customizable in the sense that the administrator can specify his or her own rules. The Management Server can be extended to manage multiple networks by receiving alerts from different SIEMs for different networks, then command and controls the appropriate NatBox, user agents and Virtual Environments.

Client Agent.

The Client Agents (also called user agents) are programs that protect production machines on which they are deployed. The Client Agents are designed to forward traffic incoming from specific machines (i.e. attackers) to the appropriate NatBox or simply deny it, following orders sent by the Management Server. Client Agents are also responsible for reporting anomalies and malfunctions, which may occur, to the Management Server. The Client Agents have a dedicated protocol to communicate with AISEN as follows

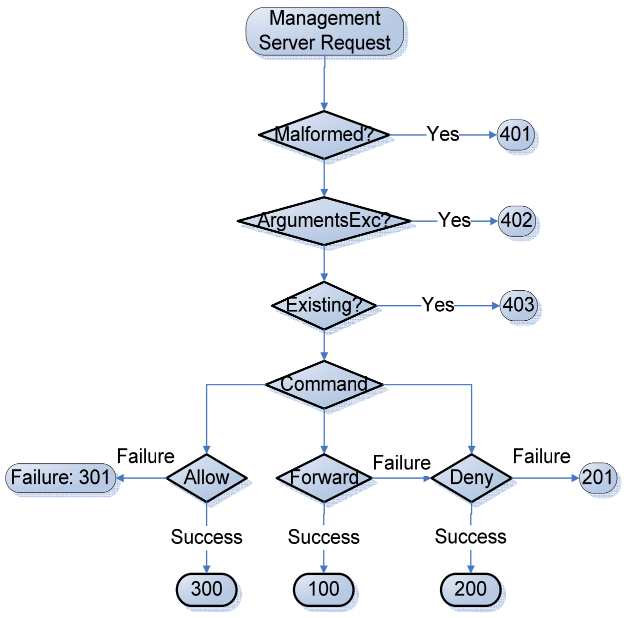

Fig. 2.Client Agent's workflow and protocol codes

NatBox.

The NatBox is a pillar piece in the overall structure of the attack- rerouting process. Its primary objective is to perform both SNAT and DNAT[9] packets flying in both directions (Attacker to honeypot and honeypot to attacker). This provides greater security as it hides the honeypot identity from the attacker and adds another controllable hop in the interfacing between the attacker and the honeypot machine. The Management Server controls the NatBox and the Client Agents with which it communicates using specified messages for receiving instructions and sending states.

The reason why NAT is not done at the level of victim machines is the lack of flexible and uniform NAT support across different windows operating systems [3][13]. Therefore, it has been decided that a number of intermediate devices (NatBoxes) would perform NAT; each of which spans across a user-specified portion of the network. In order to ensure that the solution is network independent, the NatBox should not be any of the sensitive network devices (e.g. routers or switches). It should, however, cover approximately the same area covered by a leaf router due to traffic constrains.

The interaction between the NatBox and Client Agents is show in the figure blow.

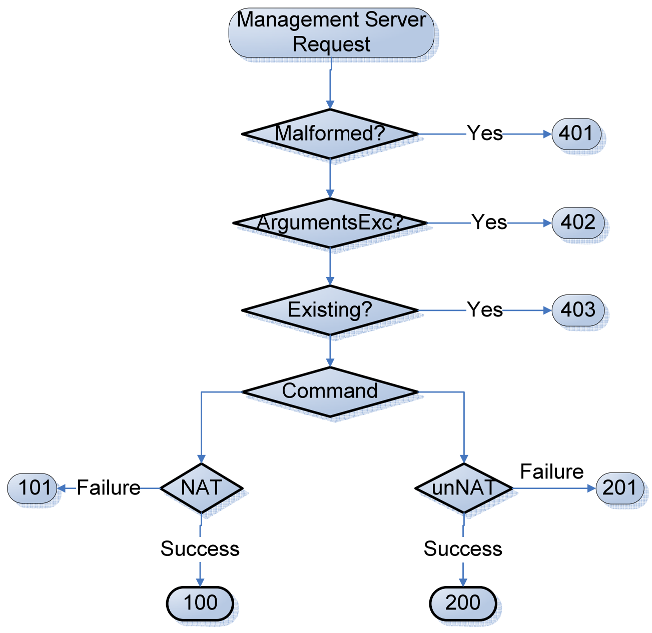

Fig. 3.NatBox's Workflow and protocol codes

Virtual Environment (VE).

The Virtual Environment refers to the set of virtual hosts created by Honeyd (Low Interaction Honeypot, defined below) as well as the virtual machines created using VMware (high interaction honeypots). Access to the Virtual Environment must be through the HoneyGate, which acts as a gateway for all incoming and outgoing traffic of the VE.

Honeyd.

Honeyd is an open source program that creates virtual hosts with customized personalities that mimic operating system and services of a real host. In our case, VEC will communicate to the Honeyd information regarding the host to mimic (OS, services). Installed on a single host, Honeyd can claim up to 65536 network addresses. This ability means that there is almost no chance of Honeyd not being able to create a virtual host upon request from the Management Server [12].

VMware Server.

VMware server will generate new virtual machines, shut down or suspend existing ones, and dynamically change their configurations upon requests from the VEC. Virtual machines, being high interaction honeypot, consume much more system resources. Consequently, VMware servers will have to follow default or administrator-based policies to optimize the use of these resources. For example, how to generate a new virtual machine in case the host running VMware server has consumed all of its resources? Possible solutions include shutting down the virtual machine that has been running the longest, idle the longest, or consuming the biggest chunk of system resources [5].

HoneyGate.

The HoneyGate is a fundamental element in the chain of command initiated by the Management Server. Essentially, the HoneyGate consists of two interdependent modules that carry out logically different tasks. These are the HoneyWall and the Virtual Environment Controller (VEC).

HoneyWall.

HoneyWall has been developed in the context of the honeynet project as a gateway device that separates high interaction honeypots from the rest of the network. This project makes similar use of honeyWall in the sense that it will act as a gateway device for both the low interaction honeypots (created using Honeyd) and the high interaction virtual machines (virtualized using VMware). In addition, HoneyWall will be responsible for capturing and logging the attacker's activities in the honeypots (data capture), containing these activities in order to minimize the risk of the attacker taking control of the honeypots (data control), and finally analyzing the captured data and converting it to useful information (data analysis) which is a major goal of this project[7].

Virtual Environment Controller (VEC) .

The VEC is a module that listens to commands from the Management Server relevant to the Virtual Environment. In fact, the VEC is responsible for coordinating the two main actors involved in honeypot generation: Honeyd and VMware Server. Once an attack is detected by the Intrusion detection system, the Management Server is notified with information about attacker and victim. The Management Server then commands the VEC to prepare the floor for the creation of a honeypot that replicates the victim's system. As a first step, the VEC commands Honeyd to create a virtual host that emulates the same operating system and services running on the real victim. However, a virtual host emulated by Honeyd is considered a low interaction honeypot that can raise suspicions about its nature. Meanwhile, VEC initiates the creation of a high interaction honeypot by commanding VMware server with corresponding parameters; including th OS and services to run on the virtual machine. The time and processing overhead required to launch a new virtual machine is one of the reasons why attacks are not directly rerouted to the virtual machine (high interaction honeypot) but are rather rerouted first to a Honeyd-based virtual host. Consequently, the VEC is also responsible for evaluating if the interaction with the low interaction honeypot is advanced enough to require the generation of a high interaction virtual machine and shift the interaction to it. This makes the VEC responsible for managing the transition from the low to the high interaction honeypot as well as the optimization of the Virtual Environment's resources[14].

III. THE REROUTING PROCESS ARCHITECTURE

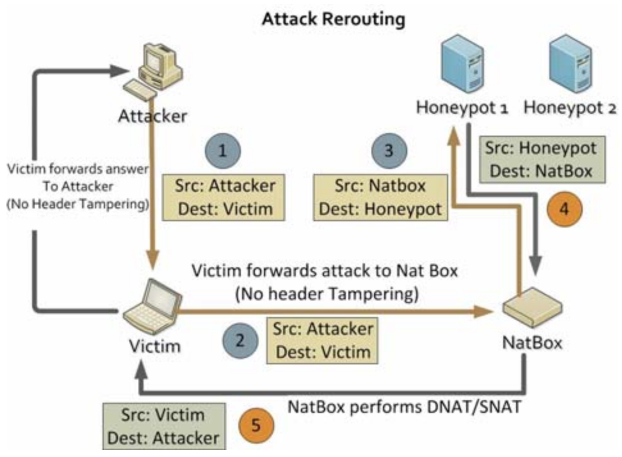
Attack Rerouting
Attacker
Honeypot 1
Honeypot 2
Victim forwards answer To Attacker (No Header Tampering)
1
Src: Attacker Dest: Victim
3
Src: Natbox Dest: Honeypot
Src: Honeypot Dest: NatBox
4
Victim forwards attack to Nat Box (No header Tampering)
Victim
2
Src: Attacker Dest: Victim
NatBox
Src: Victim Dest: Attacker
5
NatBox performs DNAT/SNAT

Fig. 4. AISEN's Rerouting Schema

A. *Rerouting Overview*

Rerouting refers to the redirection of attacks destined to victims' machines in the production environment to honeypots in the Virtual Environment. The rerouting process involves four main actors: the Client Agent, the NatBox agent, the VEC, and the Management Server. By using NatBoxes and Client Agents, the solution becomes vendor neutral. Alternatives such as modifying the routers configuration (e.g. adding entries in the routing table) are problematic as it is difficult to interface with different routers from different vendors. Moreover, changing routers configuration is difficult to troubleshoot and present higher chances of network Denial of Service when an error occurs. Moreover, attacks within a single subnet will be untreated.

NATing the traffic between the attacker and the honeypot at the level of the victim is also another alternative to AISEN's approach. However, this requires the Virtual Environment to be visible to the victims, and hence to attackers as well, which is clearly not feasible. Moreover, the victim based NATing is vendor/OS specific and lacks flexible and uniform support even across different versions of the same operating system.

The Client Agent/NatBox based rerouting is more easier to handle as the Client Agents, installed in all machines of the production environment, focus only on forwarding the suspicious traffic (with no packet header alteration) coming from the attacker to the NatBox. The ability to forward traffic is readily available in the majority of platforms. Work on optimizing the efficiency of the NatBox may be conducted later on.

As aforementioned, NatBox Agents are deployed in sensor devices (i.e. which role is to monitor network activity) that come with SIEM software. This location allows NatBoxes to be logically assigned to specific segments of the network. In a way, NatBoxes take advantage of the logical repartition of sensors in the network and will therefore receive requests only from Client Agents belonging to that network segment. Additionally, NatBoxes are the only machines able to interact with the Virtual Environment, making them hidden from the whole network. These features satisfy robust rerouting architecture as outlined below:

- *Deployment flexibility:* This rerouting scheme can be deployed in different network architectures.
- *Router Independence:* the rerouting process should not modify the router configurations. This constraint minimizes the risk of a denial-of-service (if an error occurs) and makes the process vendor independent.
- *One-to-one/Many-to-One Attacker-Victim mapping*: The rerouting process should map each attacker to a specific decoy, or all attackers targeting the same machine to the same decoy, as specified by the network administrator.
- *Troubleshooting:* The rerouting process should detect failures and errors at different levels of the process, and take appropriate decisions accordingly.
- *Encapsulation:* The Virtual Environment should be hidden from the actual network machines, and should be difficult to be detected by an attacker.

B. *Workflow Overview*

The Management Server receives Alerts mainly from the SIEM; the alert describes the Victim specifications, the attacker profile, and other important information. The Management Server then notifies the VEC to prepare a virtual host (i.e. low interaction honeypots) in the time it takes the high interaction honeypot to be ready. When the decoy is ready the Management Server notifies the NatBox to NAT the traffic coming from attacker to the victim according to the administrator preferences: One-to-One attacker-victim mapping or Many-to-One attacker-victim mapping.

When the NatBox is configured to reroute the traffic, the Management Server sends orders to the victim's Client Agent, so that it forwards all its traffic coming from the attacker to the NatBox.

The Agents in the client machines open a connection with the Management Server once it runs, and close it once it shuts down. This later one keeps track of both alive and idle machines in the Network. When an agent is Idle, the Management Server commands the VEC to create a temporary Honeyd virtual host and orders the appropriate NatBox to reroute the traffic destined to the Idle Victim to the Honeyd. When the Honeyd virtual host gets an important incoming traffic, it sends an alarm to the Management Server as a high alert. If there is no generated warning from the IPS/IDS, the data collected will have a high chance to be a zero-day attack. Henceforth, the Management Server commands the VEC to prepare a high-interaction honeypot to collect data.

IV. CONCLUSION

Network security has always been considered one the most critical aspects of today's businesses. The time and effort put in place to provide information confidentiality, integrity, and availability is growing exponentially. However, as networks become bigger and more complex and attacks become more sophisticated, along with exponential growth of vulnerabilities found, traditional security measure are turning inefficient and, as a result, ineffective. [14, 15, 16, 17]

This paper explained how AISEN adds a layer of autonomy to already secured networks. This solution ensures integrity, scalability, and robustness, using a state-of-the-art rerouting architecture combined with a hybrid use of both low and high interaction honeypots. The need for human intervention has always been an issue as there are many factors that affect our performance such as unavailability, long response time, and human error. Network monitoring tools, automatic honeypot generation and malicious traffic rerouting along with AISEN's critical features will help make the security administrator's job easier by limiting them to the status of observer and decision maker, and helping the CERT/CSIRT teams gather evidences easily.

References

[1] Jiang, X., Xinyuan,W.Out-of-the-box Monitoring of VM-based High-Interaction Honeypots. Dissertation,George Mason University.(2007)

[2] Revolution Systems (2010).Linux NAT in Four Steps Using Iptables.http://www.revsys.com/writings/quicktips/nat.html. Accessed 11 October 2011.

[3] González, D.Installing a Virtual HoneyWall Using VMware (2004).In: Spanish Honeynet Project. Available viaPapers. http://honeynet.org.es/papers/vhwall. Accessed 11 October 2011.

[4] The Honeynet Project (2008). Configuring VMware and Installing Your Honeypots. http://www.honeynet.pk/honeywall/eeyore/page2.htm. Accessed 11 October 2011.

[5] Symantec (2010). Open Source Honeypots, Part Two: Deploying Honeyd in the Wild. http://www.symantec.com/connect/articles/open-source-honeypots-part-two-deploying-honeyd-wild. Accessed 11 October 2011.

[6] The Honeynet Project (2008). The HoneyWall.http://www.honeynet.org. Accessed 07 May 2011.

[7] Microsoft Technet(2003). Defining Malware: FAQ. http://technet.microsoft.com/en-us/library/dd632948.aspx. Accessed on 11 October 2011.

[8] The Internet Engineering Task Force(1994). RFC 1631 - The IP Network Address Translator (NAT).http://tools.ietf.org/html/rfc1631. Accessed on 11 October 2011.

[9] Lane A (2010). Understanding and Selecting SIEM/LM: Use Cases.In: Securosis Blog. Securosis. Available via http://securosis.com/blog/understanding-and-selecting-siem-lm-use-cases-part-1. Accessed 11 October 2011.

[10] Hudak S (2008). Automatic Honeypot Generation and Network Deception. In: Scientific Literature Digital Library and Search. Accessed on 12 July 2011.

[11] ProvosN (2003). A Virtual Honeypot Framework. University of Michigan. Accessed on 4September2011.

[12] CyberCiti (2010). Mac OS X: Set Port Forwarding Nat Router (Internet Sharing). http://www.cyberciti.biz/faq/howto-configure-macosx-as-nat-router. Accessed on 12 October 2011.

[13] Hecker C, Kara N, and Brian H (2006). Dynamic Honeypot Construction. University of Alaska Fairbanks. Accessed 11 October 2011.

[14] Hakimeh Alemi Baktash, Mohammad Bagher Karimi, Mohammad Reza Meybodi, Asgarali Bouyer . A new access control framework in Grid Environments. Journal of Networking Technology, 2 (1): pp 36-43. (2011).

[15] Blesson Varghese, Gerard McKee, Vassil Alexandrov. Triad Views of Cognition in Intelligent Agents and Intelligent Cores for Fault Tolerance, Journal of Intelligent Computing, 1 (1): pp. 40-47 (2010).

[16] Brandon Rohrer. Concept Acquisition for Dialog Agents. Journal of Digital Information Management, 1 (3): pp. 189-201. (2010).

[17] Kazunari Ishida. Spatial Analysis on Social Media-Development of Location-based Contents System with Mobile Devices. Journal of Networking Technology, 2 (4); pp. 146-161. (2011).