An Ontological Approach to Secure Address Resolution Protocol

Lecturer Dr. Enas Hadi Salih*

Abstract

Address Resolution Protocol is an essential protocol deployed over the network session where it maps logical addresses (IP- Internet Protocol) to physical addresses (MAC-Media Access Control).

In this paper a multi-agent system is introduced and implemented to prevent known attacks against this protocol (i.e., ARP spoofing and Sniffing) by constructing a cognitive upper layer that sustains the validity and the authenticity of ARP traffic.

The presented system is a preemptive system that utilizes domain intelligence to reveal smart attacking methodologies against Spoofing terminologies; the presented system is implemented JADE environment and low level software modules. Results are obtained and analyzed to verify the effect of the approach.

Keywords: Ontology, ARP, network security, JAVA Agent, JADE environment, ARP spoofing.

*Al-Rafidain University College

1. Introduction

One of the biggest threats in a computer network is a rogue system pretending to be a trusted host. Once someone has successfully impersonated another host, he/she can intercept and log traffic destined for the real host, or lie in wait for clients to connect and begin sending host confidential information.[1,2]

Spoofing a host has especially severe consequences in IP networks, because it opens many other avenues of attack. One technique for spoofing a host on an IP network is Address Resolution Protocol (ARP) spoofing. ARP spoofing is limited only to local segments and works by exploiting the way IP address are translated to hardware Ethernet addresses.[1,3,4]

When an IP datagram is sent from one host to another on the same physical segment, the IP address of the destination host must be translated into a MAC address. This is the hardware address of the Ethernet card that is physically connected to the network. To accomplish this, the Address Resolution Protocol is used.[1,3,4]

ARP is a stateless protocol which means accepting responses by hosts connected to the network without sending any request, this is a tremendous weak point exploited by attackers who want to receive traffic destined for another host, attacker could send forged ARP responses that match chosen IP address with attacker MAC address. The machine that receive these spoofed ARP responses can't distinguish them from legitimate ARP responses and will begin sending packets to the attacker's MAC address.[1,3,4,5]

Another side effect of ARP being stateless is that a system's ARP tables usually use only the results of the last response. In order for someone to continue to spoof an IP address, it is necessary to flood the host with ARP responses that overwrite legitimate ARP responses from the original host. This particular kind of attack is commonly known as ARP cache poisoning.[1,3,4]

2.Network Sniffing using ARP spoofing (Theoretical Background)

In a *switched network environment*, packets are only sent to the port they are destined to, according to their destination MAC addresses. This requires more intelligent hardware that can create and maintain a table associating MAC addresses with certain ports, depending on which device is connected to each port, as illustrated in figure (1):[4,5,6]

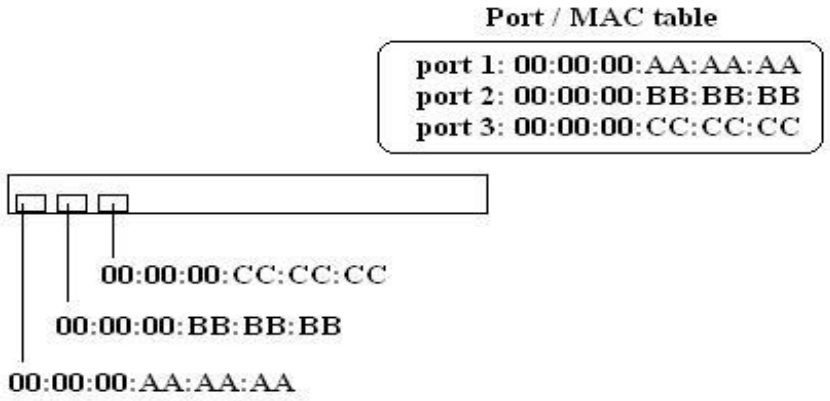


Figure 1: Mapping hardware port to host physical address

The advantage of a switched environment is that devices are only

sent packets that are meant for them, meaning that promiscuous devices aren't able to sniff any additional packets. But even in a switched environment, there are clever ways to sniff other devices' packets; they just tend to be a bit more complex. In order to find hacks like these, the details of the protocols must be examined and then combined.[4]

One important detail of network communications that can be manipulated for interesting effects is the source address. There's no provision in these protocols to ensure that the source address in a packet really is the address of the source machine. The act of forging a source

address in a packet is known as *spoofing*. The addition of spoofing to the bag of tricks greatly increases the number of possible hacks, because most systems expect the source address to be valid.[4]

Spoofing is the first step in sniffing packets on a switched network. The other two interesting details are found in ARP. First, when an ARP reply comes in with an IP address that already exists in the ARP cache, the receiving system will overwrite the prior MAC address information with the new information found in the reply (unless that entry in the ARP cache was explicitly marked as permanent). The second detail of ARP is that systems will accept an ARP reply even if they didn't send out an ARP request. This is because state information about the ARP traffic isn't kept, because this would require additional memory and would complicate a protocol that is meant to be simple.[4]

These three details, when exploited properly, can allow an attacker to sniff network traffic on a switched network with a technique known as *ARP redirection*. The attacker sends spoofed ARP replies to certain devices that cause the ARP cache entries to be overwritten with the attacker's data. This technique is called *ARP cache poisoning*. In order to sniff network traffic between two points, *A* and *B*, the attacker needs to poison the ARP cache of *A* to cause *A* to believe that *B*'s IP address is at the attacker's MAC address, and also poison the ARP cache of *B* to cause *B* to believe that *A*'s IP address is also at the attacker's MAC address. Then the attacker's machine simply needs to forward these packets to their appropriate final destinations, and all of the traffic between *A* and *B* still gets delivered, but it all flows through the attacker's machine, as shown in figure (2):[4]

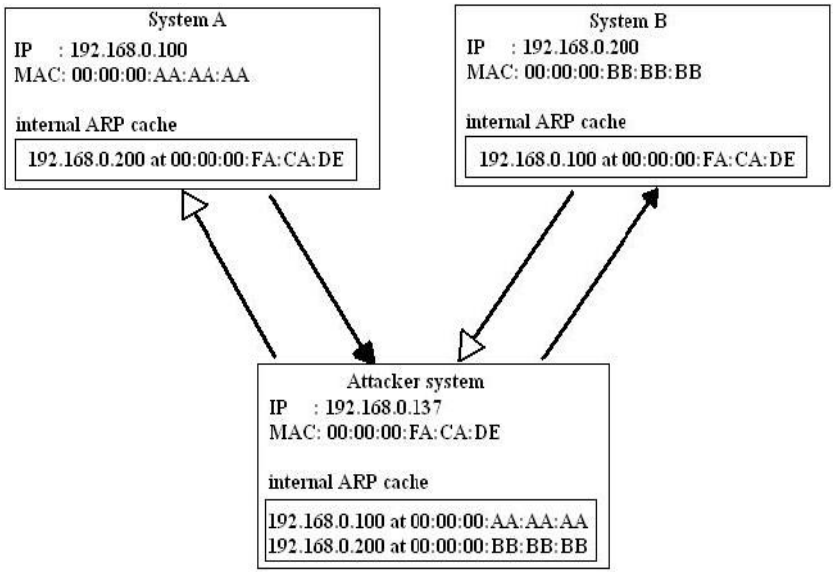


Figure 2 : ARP spoofing scenario

Because *A* and *B* are wrapping their own Ethernet headers on their packets based on their respective ARP caches, *A*'s IP traffic meant for *B* is actually sent to the attacker's MAC address, and vice versa. The switch only filters traffic based on MAC address, so the switch will work as it's designed to, sending *A*'s and *B*'s IP traffic, destined for the attacker's MAC address, to the attacker's port. Then the attacker rewraps the IP packets with the proper Ethernet headers and sends them back out to the switch, where they are finally routed to their proper destination. The switch works properly; it's the victim machines that are tricked into redirecting their traffic through the attacker's machine.[4]

Due to time-out values, the victim machines will periodically send out real ARP requests and receive real ARP replies in response. In order to maintain the redirection attack, the attacker must keep the victim machine's ARP caches poisoned. A simple way to accomplish this is to simply send spoofed ARP replies to both A and B at a constant interval, perhaps every ten seconds.[4]

A *gateway* is a system that routes all the traffic from a local network out to the Internet. ARP redirection is particularly interesting when one of the

victim machines is the default gateway, because the traffic between the default gateway and another system is that system's Internet traffic. For example, if a machine at 192.168.0.118 is communicating with the gateway at 192.168.0.1 over a switch, the traffic will be restricted by MAC address. This means that this traffic cannot normally be sniffed, even in promiscuous mode. In order to sniff this traffic, it must be redirected.[4]

3.Software Agent

The term ‘agent’, or software agent, has found its way into a number of technologies and has been widely used, for example, in artificial intelligence, databases, operating systems and computer networks literature. Although there is no single definition of an agent, all definitions agree that an agent is essentially a special software component that has autonomy that provides an interoperable interface to an arbitrary system and/or behaves like a human agent, working for some clients in pursuit of its own agenda. Even if an agent system can be based on a solitary agent working within an environment and if necessary interacting with its users, usually they consist of multiple agents. These multi-agent systems (MAS) can model complex systems and introduce the possibility of agents having common or conflicting goals. These agents may interact with each other both indirectly (by acting on the environment) or directly (via communication and negotiation). Agents may decide to cooperate for mutual benefit or may compete to serve their own interests.[7,8]

4.Java Agent DEvelopment(JADE)

The first software developments, that eventually became the JADE platform, were started by Telecom Italia (formerly CSELT) in late 1998, motivated by the need to validate the early FIPA specifications.[8]

Partially funded by European Commission (FACTS project, ACTS AC317) a team composed of Fabio Bellifemine, Agostino Poggi and Giovanni Rimassa were gathered with the good will and dedications to promote the concepts of JADE and its compliant to FIPA. At a certain point it was decided to move beyond a means of simply validating the FIPA specifications towards developing a fully fledged middleware platform. The vision was to provide services to application developers and that were readily accessible and usable by both seasoned developers and

newcomers with little or no knowledge of the FIPA specifications. Emphasis was placed on the simplicity and usability of the software APIs.[8,9]

In order to better facilitate industrial involvement, in May 2003 Telecom Italia Lab and Motorola Inc. defined a collaboration agreement and formed the JADE Governing Board, a not-for-profit organization of companies committed to contributing to the development and promotion of JADE. The Board was formed as a contractual consortium with well-defined rules governing the rights and obligations toward generated IPR. The Board is open with members able to join and leave according to their needs. At the time of writing, Telecom Italia, Motorola, France Telecom R&D, Whitestein Technologies AG and Profactor GmbH have all become members of the Board.[8,9]

When JADE was first made public by Telecom Italia, it was used almost exclusively by the FIPA community but as its feature set grew far beyond the FIPA specifications, so did its usage by a globally distributed developer community. It is interesting to note that JADE contributed to widespread diffusion of the FIPA specifications by providing a set of software abstractions and tools that hid the specifications themselves; programmers could essentially implement according to the specifications without the need to study them. This is considered as one of the main strengths of JADE with respect to FIPA.[8,9]

5.Agent-Based ARP Authentication (The Proposed System)

This proposal is dedicated to provide a promotion step toward fighting network attackers by leveraging up the methodology from raw data to conceptual domain, where problem domain is to be abstracted into its essential concepts and ontology is to be defined over that domain. Intelligent Agent has been deployed to perceive domain concepts and makes decisions upon its conceptual model.

Intelligent Agent has been designed and implemented over this work to autonomously authenticate the ARP packet before replying it, this approach needed collaborative components at different level to work together, as it is shown in figure (3), where Intelligent Agent mobility imposes the implementation to be in JAVA but with a sacrifice for the

accessibility to kernel level operating system resources, thus socket layer has been added to grant JAVA the ability to communicate C++ module which is eventually communicate kernel level driver.

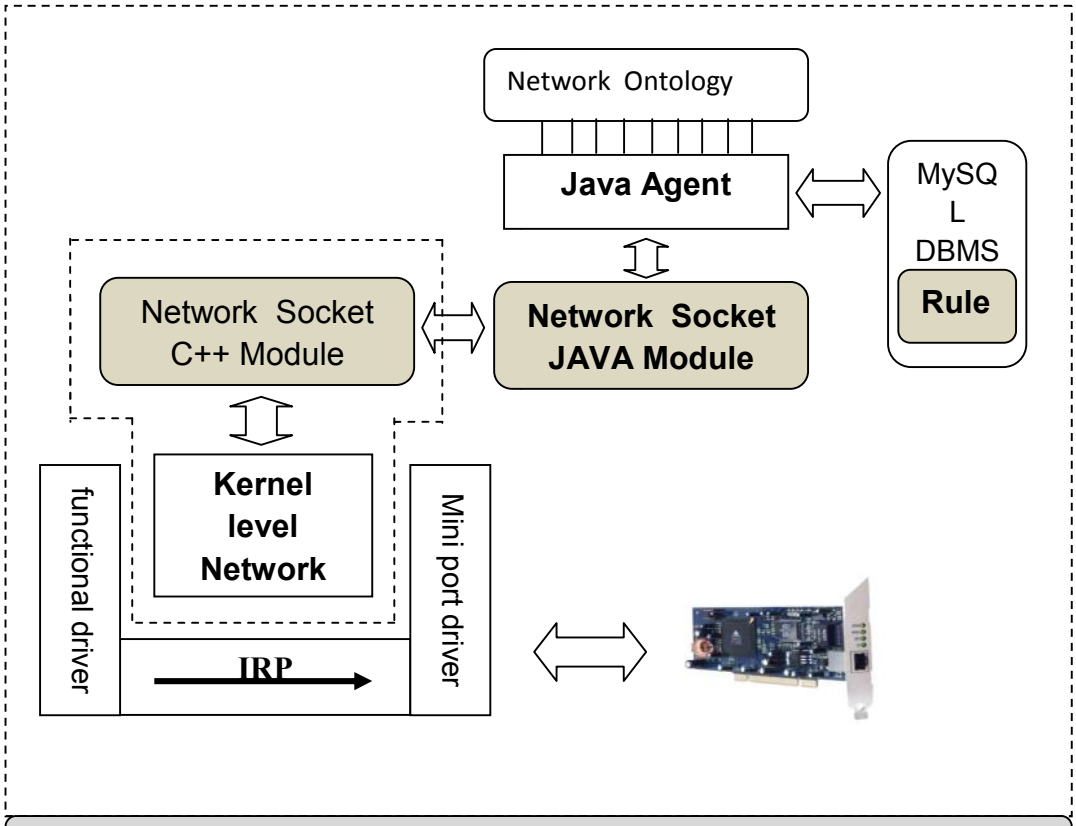


Figure 3- Proposed System Essential Components

To grant this proposal the reliability needed to authenticate network packets (ARP) before replying it through operating system components, the proposed technique contains network filter driver; this driver has been designed using Microsoft DDK combined with Visual studio 6.0 to intercept network traffic, carried by IRP (I/O Request Packet), before getting into host kernels.

Figure (4) presents the conceptual view for the Agent structure where its functionality can be abstracted into two essential behaviours; one for authenticate ARP (forward ARP) and RARP (reverse ARP)

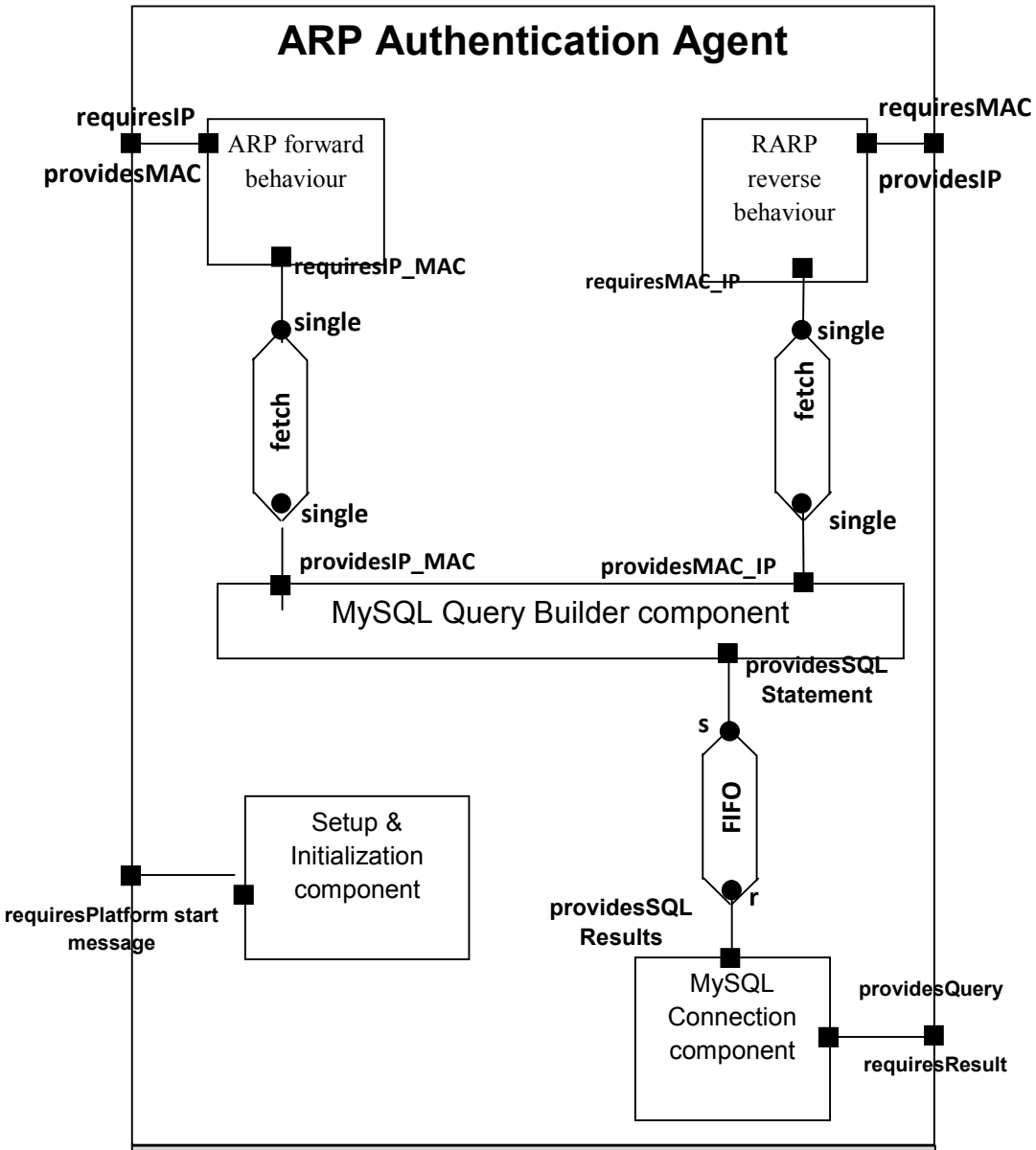


Figure 4: ARP Authentication Agent – Conceptual view

Along this work, ARP ontology has been defined to transfer network ARP protocol to conceptual domain. Next step was to represent this ontology in format easily can be interpreted, thus, XML format has been chosen to represent ARP ontology.

Basically, ARP protocol is a mapping function from IP domain to MAC domain; this is the forward mapping (ARP session) and from MAC domain to IP domain; this is the reverse mapping (RARP session).

Figures (5) and figure (6) represent ARP session and RARP session; also the XML representation is presented. XML will be encapsulated into ACL message crossing Agent platform, it is Agent responsibility to interpret message contents and acts according to that interpretation.

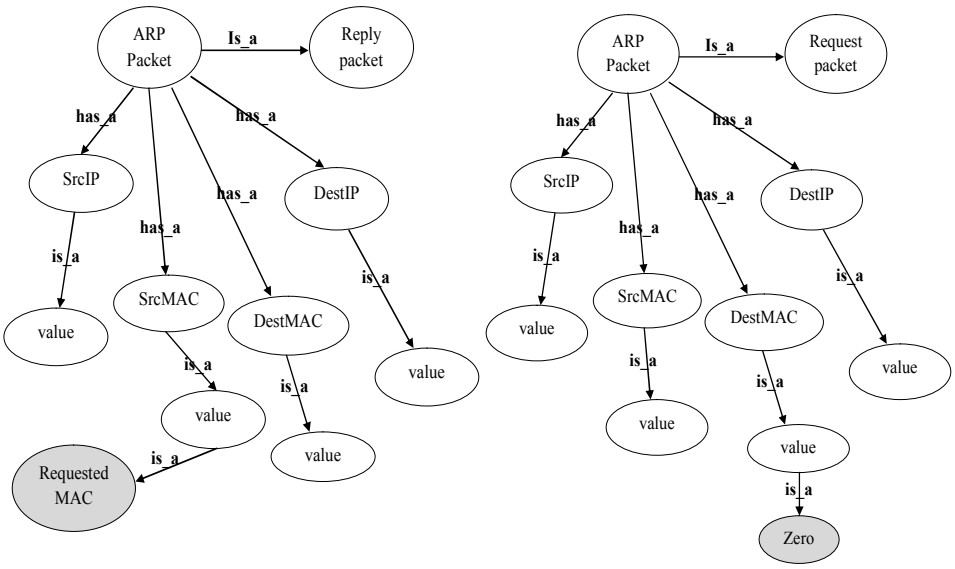
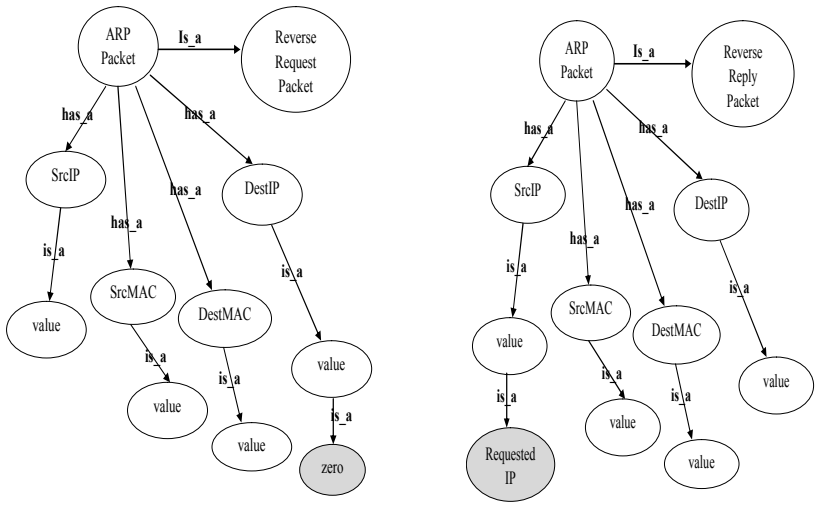


Figure 5: ARP forward session, Ontology and XML representation




```
<ARP_Packet type ="RREQUEST"> <SrcIP> value = "specified " </SrcIP> <SrcMAC> value = "specified" </SrcMAC> <DestIP> value = "requested" </DestIP> <DestMAC> value = "specified" </DestMAC> </ARP_Packet>
```

```
<ARP_Packet type ="RREPLY"> <SrcIP> value = "declared " </SrcIP> <SrcMAC> value = "specified" </SrcMAC> <DestIP> value = "specifed" </DestIP> <DestMAC> value = "specified" </DestMAC> </ARP_Packet>
```

Figure 6: RARP reverse session , Ontology and XML representation

In a testing environment, the network filter driver will intercept ARP session presented in figure (7), which is a screenshot of Wireshark the network sniffing and analysis tool.

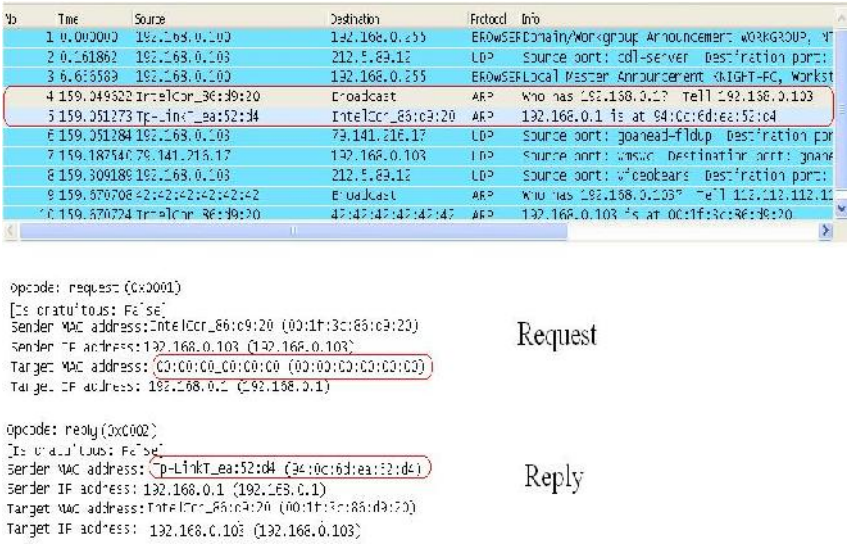


Figure 7: Wireshark software capturing ARP session at network level

Windows operating system is following the same scheme in building ARP cache table for the testing environment. Figure (8) shows ARP cache table built by windows operating system.

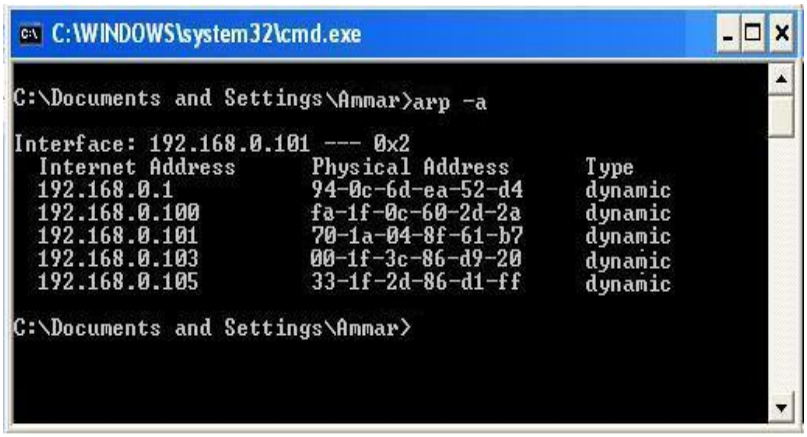


Figure 8: conventional ARP cache table

This proposal system will built analogous ARP cache table as shown in figure (9); this table is Agent-based where another column has been added to traditional ARP cache presented in figure (8).

index	AgentID	IP	MAC
1	ArpAgent001@192.168.0.100	192.168.0.100	fa-1f-0c-60-2d-2a
2	ArpAgent002@192.168.0.100	192.168.0.101	70-1a-04-8f-61-b7
3	ArpAgent003@192.168.0.100	192.168.0.103	00-1f-3c-86-d9-20
4	ArpAgent004@192.168.0.100	192.168.0.105	33-1f-2d-86-d1-ff

Figure 9: Agent-Based ARP cache table

Agent-based ARP cache table will saved at each host and updated upon firing new agents on new hosts. Each host now will resolve for AgentID rather than for IP or MAC, thus the attacker will face a new obstacle that is to prove himself as authenticated agent within the network.

Further more, all agents within the network will be monitored and managed through platform monitoring tools come with developing envrionment. Figure (10) shows RMA (Remote Management Agent) tool used by this proposal to monitor the creation and termination of authenticated agents.

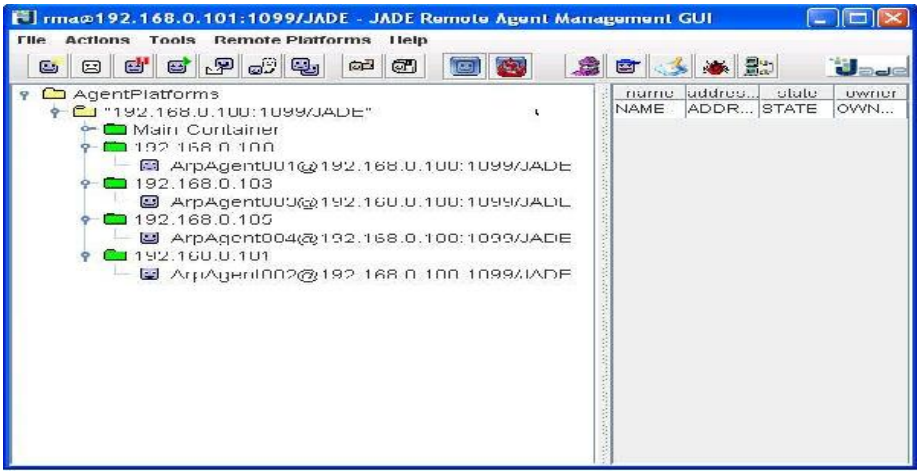


Figure 10: Remote Agent Management for ARP Agents

When ARP Agent perceived the network environment according to the ontology defined in figure (5) and figure (6), thus when a broadcast ARP packet is received the agent will retrieve the Agent source for the that packet and communicate it at Agent level to authenticated this packet before completing ARP session.

ARP Agent will use ACL (Agent Communication Language) messages to communicate other agents within the platform, and the message content will be the XML representation for the ontology designed and implemented by this proposal. Figure (11) shows a complete ARP session accomplished at Agent level, message content is shown as message types are also highlighted.

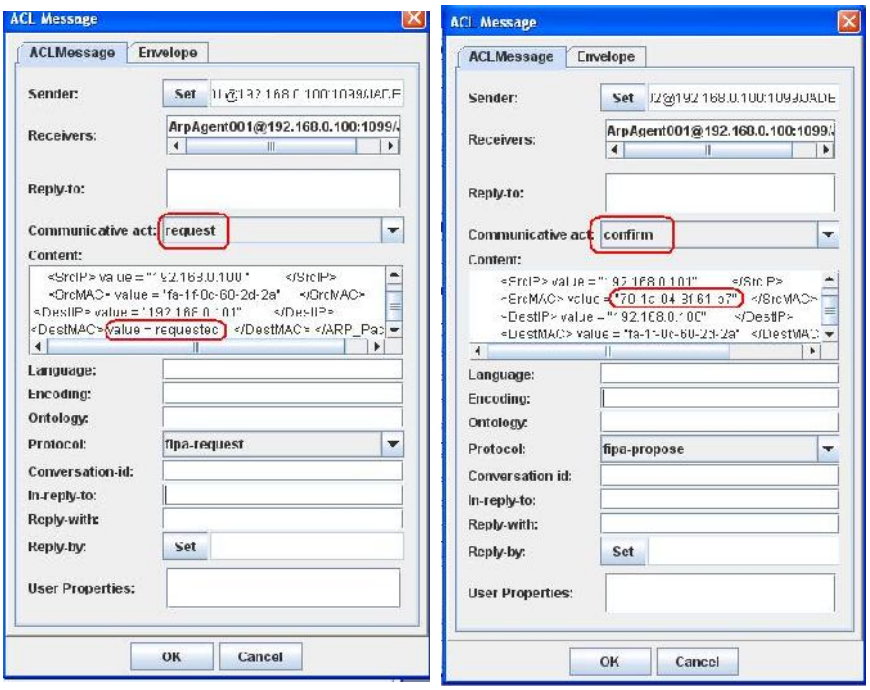


Figure 11: ARP forward session at Agent level and using XML based ontology

6.Conclusions

1- Agent software module can be embedded in perceivable domains to add the ability to make decisions where sensor and raising alerts can be the assignments of agents. Agent platform can be built to assist sustaining events occurred at low level abstraction of any information system

2- Counter network threats can be implemented as a social activity rather than personal inspection for the threat. all network threats have their signature through which security software can identify and neutralize these threats, thus, if for some reason certain threat's signature is not recognized in some network segment; it could be recognized somewhere else.

3- Application level proxies can make a great use of intelligent agent in the domain of the application, where a domain can be abstracted in its essential concepts and semantics. This paper exploited ARP concepts to increase Agent cognitive perceiving of the problem domain.

4- Network protocols are easy to be conceptualized due to its design principles in maintaining the simplicity and lower network traffic, and ethical hacking methodologies are producing many concepts to identify threat signature and footprint. Powerful network ontology will grant security products to conceptualize network events and eventually identify not only known threats' signature but even the anomalous behaviors.

References

1- Anderw lockhart, " Network security hacks , Tips & Tools for Protecting Your Privacy ", O'Reilly, USA, 2007
2- William Stallings, " Cryptography and Network Security , principles and practices ",fourth edition, Pearson Prentice Hall, 2006
3- Sergio Scaglia, " The Embedded Internet, TCP/IP basics, implementation and applications", Addison Sesley, USA, 2007.
4- Jon Erickson , " Hacking: The Art of Exploitation ",William Pollock, USA, 2003.
5- Brian Caswell and Jay Beale, " Snort 2.1 Intrusion Detection", second edition, Syngress publishing, Inc. 2004.
6- Carl Endorf, Eugene Schultz, and Jim Mellander,"Intrusion Detection and Prevention", USA, McGraw-Hill, 2004.
7- Fabio Bellifemine, Giovanni Caire and Dominic Greenwood,"Developing multi-agent systems with JADE", John Wiley & Sons, Ltd, 2007.
8- Fabio Bellifemine, Giovanni Caire, Tiziana Tucco and Giovanni Rimassa, " JADE Progammer's Guide", TILab S.P.A, 2010.
9- Rafael H. Bordini, "Multi-Agent Programming: Languages, Platforms and Applications", Springer, USA, 2005

أمنية الشبكات: نهج أدراكي لحماية بروتوكول حل العنوان

م.د.أيناس هادي صالح*

المستخلص

ان بروتوكول حل العنوان هو بروتوكول رئيسي في جلسة الشبكة حيث يقوم بالربط بين العنوان المنطقي (IP- Internet Protocol) والعنوان الفيزيائي للاجهزة المربوطة على الشبكة (MAC- Media Access Control).

في هذا البحث تم تقديم وتنفيذ نظام حماية من الهجمات المعروفة ضد هذا البروتوكول (ARP Spoofing and Sniffing) وذلك من خلال بناء نظام معرفي يعمل في مستوى التطبيق حسب تعريف مستويات الاتصال في الشبكات والذي يقوم بتحقيق مصداقية بروتوكول ARP من خلال توفير طبقة حماية ذكية تطور معرفتها عن مصادر الشبكة خلال العمل.

النظام المقترح هو نظام وقائي والذي يستخدم الذكاء الموجود في مجال التطبيق لكشف الهجمات الذكية والتي تستهدف بروتوكول ARP ، النظام المقترح تم تنفيذه باستخدام بيئة بناء الوكلاء JADE وبرامجيات اخرى مساندة على مستوى النظام وخلال التنفيذ تم جمع النتائج وتحليلها للتحقق من تاثير هذا النهج حيث قام النظام المقترح باكتشاف هجوم نجح على مستوى البيانات ولكنه فشل في خداع النظام.

* كلية الرافدين الجامعة