Authentication services in mobile ad hoc networks

Willy Ronald Jimenez Freitez, Hakima Chaouchi, Maryline Laurent

Authentication Services in Mobile Ad-hoc Networks

LOgiciels-**R**éseaux	**Willy Jiménez** **Hakima Chaouchi** **Maryline Laurent-Maknavicius**	08013 -LOR

Authentication Services in Mobile Ad-hoc Networks

ABSTRACT

The deployment of wireless ad hoc networks is useful for people when they desire to communicate even if they are not connected to any infrastructure, with the purpose of playing games, sharing internet connection, or exchange files.
In some ad hoc scenarios, they might know each other, so they can establish trusted relationships. However, if the number or users and mobility increase then it is more complicated to trust all users and a security mechanism is required. Few researches has been done in this field to find security solutions for MANETs deployments; one of them proposes a framework where the traditional AAA services are distributed inside the network with the idea of allowing secure exchange of services that could be chargeable.
Based on this framework, we evaluate technical solutions, focusing mainly on the Authentication service; in order to have real implementations. One possibility is using virtualization technology to offer a de-centralized authentication service. Another solution is the development of a secure version of a routing protocol that uses a de-centralized authentication service as a previous requirement to allow any node to join the ad hoc routing domain.

Willy Jiménez	Hakima Chaouchi	Maryline Laurent-Maknavicius
Etudiant	Maître de Conférences	Professeur
TELECOM & Management SudParis. Département LOR	TELECOM & Management SudParis. Département LOR	TELECOM & Management SudParis. Département LOR
9, rue Charles Fourier	9, rue Charles Fourier	9, rue Charles Fourier
91011 Evry cedex	91011 Evry cedex	91011Evry cedex
E-mail: willy.jimenez_freitez@	E-mail: hakima.chaouchi@	E-mail: Maryline.Maknavicius@
telecom-sudparis.eu	it-sudparis.eu	it-sudparis.eu

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Internet has become crucial in modern business world activities nowadays; the availability of different types of applications has created a high demand for services with a permanent availability requirement (always on) in an environment with users' mobility where laptops are the preferred instrument used to access such services because of their wireless connections.

The deployment of wireless LAN networks in strategic places is becoming very common in big cities, hotels, airports and others with the goal of attracting people for tourism or business. However most of these networks are somehow attached to a wired network in order to control the access to the network and authenticate, authorize and charge legal users who paid to get the service.

Another possibility that have not been widely exploited is to generate autonomous wireless networks using the ad hoc connection capabilities of the wireless cards. In such environments all the nodes are connected together and if the number of nodes and the area it covers increases them a routing protocol is needed. Usually, these ad hoc networks are mainly used to share files and internet connections because there are not still business models that could take advantage of this kind of networks, and one of the main reasons is the lack of AAA services due to the distributed nature of these networks.

However there is a possibility of having these services inside the ad hoc network, where an operator could pre-configure some nodes with AAA services and use them to deploy an ad hoc network and make business. The potential users have to be authenticated before having access to services and their operations will be registered for charging purposes. Then the challenge is how to deploy this kind of networks.

The present report describes two possible solutions to the problem described above: using virtual machines and/or develop a secure version of a routing protocol. The idea behind virtual machines is to have preconfigured nodes which have a main operating system where sensitive AAA services are configured, while the service or application to be offered to the users is them installed in a virtual environment. Additionally these nodes could be configured as access points and could create a mesh network.

Regarding the second solution, a modification to secure OLSR protocol is proposed and implemented, where we add a new behavior such that the nodes participating in the routing domain are authenticated against a radius server in order to participate in the routing domain. The practical implementation was done modifying the secure plugin of OLSR.

The rest of the report is organized as follow: Chapter 2 there is a review of wireless LAN (WLAN) networks to introduce mobile ad hoc networks (MANET) topology, routing protocols and possible attacks. Chapter 3, the WLAN security is studied, focusing mainly on authentication security. In chapter 4 we considered a framework that proposed the implementation of distributed authentication services inside MANETs to grant security and

access to services. Based on this work, chapter 5 describes a possible practical solution using virtualization technologies. In chapter 6 we propose an alternative solution that modifies the behavior of secure OLSR protocol in order to include an authentication step. The solution was implemented modifying the secure olsrd plugin. Finally, we present our conclusions.

2 WIRELESS LOCAL AREA NETWORKS (WLAN)

Wireless LAN technology has become a popular solution to offer services to users with certain mobility since they allow connections to local area networks LANs without cables. WLANs are standardized by IEEE under the standard 802.11x and are known by its common name: WiFi. They can work mainly at rates of 11Mbps or 54Mbps in the frequency band of 2.4 GHz and the standardization guarantees interoperability among different WiFi devices and a high availability of wireless cards and devices for laptops, though laptops had integrated wireless cards and even other types of terminals like PDAs and mobile phones has WiFi.

WLANs can be built in two main network configurations:

- Infrastructure mode or networking with a base station or access point (see figure 1). Here users have to connect to the access point, which is then cabled to a wired network. The nodes cannot communicate directly to each other, instead they have to communicate through the AP. Common business application for this configuration are hotspots and mesh networks. In both cases these networks depend on wire infrastructure which provides the authentication, authorization and accounting services.

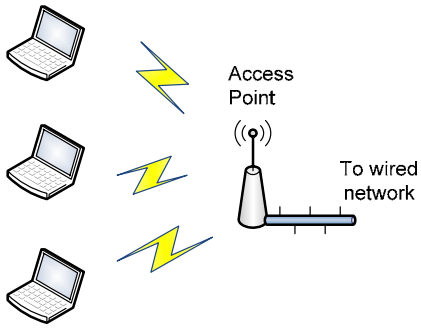

Figure 1: Infrastructure Wireless Network

- Ad-hoc mode (see figure 2). In such mode all the equipments or terminals are connected together without any infrastructure, in this way people can share files and even the internet connections without AP.

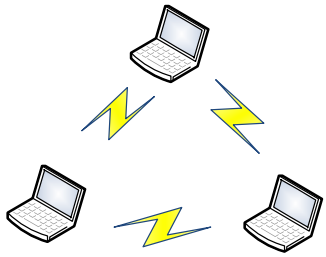

Figure 2: Ad hoc Wireless Network

2.1 Mobile Ad-hoc Networks (MANETs)

In the case that ad hoc nodes have some arbitrary mobility in the network then we have a Mobile Ad hoc NETwork or MANET. A MANET is formed on demand and all nodes are interconnected covering certain geographical area. Some scenarios suitable for MANET are: disaster recovery, military applications, and people meetings.

According to RFC2501 MANETs have the following characteristics:

- Dynamic topologies.
- Bandwidth constrains.
- Energy constrains.
- Limited physical security.

Routing service in a MANET is important because there is no infrastructure available, thus nodes have to act like routers to forward packets and also manage the dynamicity of the topology. If nodes cannot act like routers then communication among them is only possible if they are in the radio communication range. In the presence of nodes that are willing to act as mobile routers where they route or forward information; they form a multihop MANET.

2.1.1 Routing Protocols for Ad hoc Networks

We have basically two types of routing protocols, proactive and reactive. In proactive routing the routers maintain routes to several destinations in the network and the information about the network topology is published periodically. The purposed is to have a valid path to any node when a node will send a packet. However this methodology generates overhead in the network and consumes nodes power. They are good for limited number of nodes and mobility. Examples of this category are the Dynamic Destination Sequenced Distance Vector Routing protocol (DSDV) and Optimized Link State Routing protocol (OLSR).

In reactive routing, routers search for the route only when is needed. It consumes fewer resources but transmission time is bigger because the packet has to wait for the establishment of the route before being transmitted. They are good in the case where there is a large number of nodes and high mobility of the nodes. Examples of this category are the Dynamic Source Routing protocol (DSR) and Ad hoc On-demand Distance Vector routing protocol (AODV).

Additionally, there are hybrid protocols which can use both reactive and proactive routing according to the network conditions such as the Zone Routing Protocol (ZRP).

3 WLAN SECURITY

The radio air is the transmission media in WLAN and all communications are broadcasted, thus anyone with the appropriate tools can listen to the transmissions, sniff them and compromise the security of the network. By security of the network we refer to availability of the service, access control to the service, information integrity, data and device authentication and data confidentiality.

Wired Equivalent Privacy (WEP) was one of the first efforts to bring security to WLAN, limiting the access to the wireless media. This protocol uses a symmetrical cipher algorithm RC4 in order to provide access control, confidentiality, authentication and integrity. However it became highly vulnerable because the key used to cipher is weak and can be easily derived from the algorithm; in consequence a WEP connection can be cracked within some minutes using programs available on Internet. Other security protocols were designed such as WPA, and WPA2.

This situation generated new proposals and solutions in order to have security in WLAN; as for example the Extensible Authentication Protocol EAP. This protocol does not use any particular authentication mechanism; instead it allows the peers to negotiate using EAP in the connection authentication phase and if it is successful the peers then negotiate the use of a specific EAP authentication scheme, better known as an EAP method.

3.1 Authentication in WLAN

The norm 802.1X was initially known by the security management of wired switched packet networks, it established a Port Based Network Access Control which allows or block the data flow of an unknown user. The architecture of the standard is composed by a supplicant, an authenticator and the authentication server.

802.1X defines a standard for encapsulating the Extensible Authentication Protocol (EAP) messages so that they can be handled directly by a LAN MAC service. This encapsulated form of EAP frame is known as EAPOL. Additionally EAPOL also provides control functions such as start, logoff, and key distribution. Only EAP frames are allowed during the unauthorized state, after authorization, the data flows are permitted.

In wireless applications the scenario is composed of the supplicant, a wireless terminal that wants access to the network; the authenticator, a wireless access point that is connected usually by cable to the authentication server which is typically a RADIUS server (RFC2865). The port concept is replaced by the association between the supplicant and the access point. In figure 3 we can see its basic architecture and the message exchange performed during the authentication process.

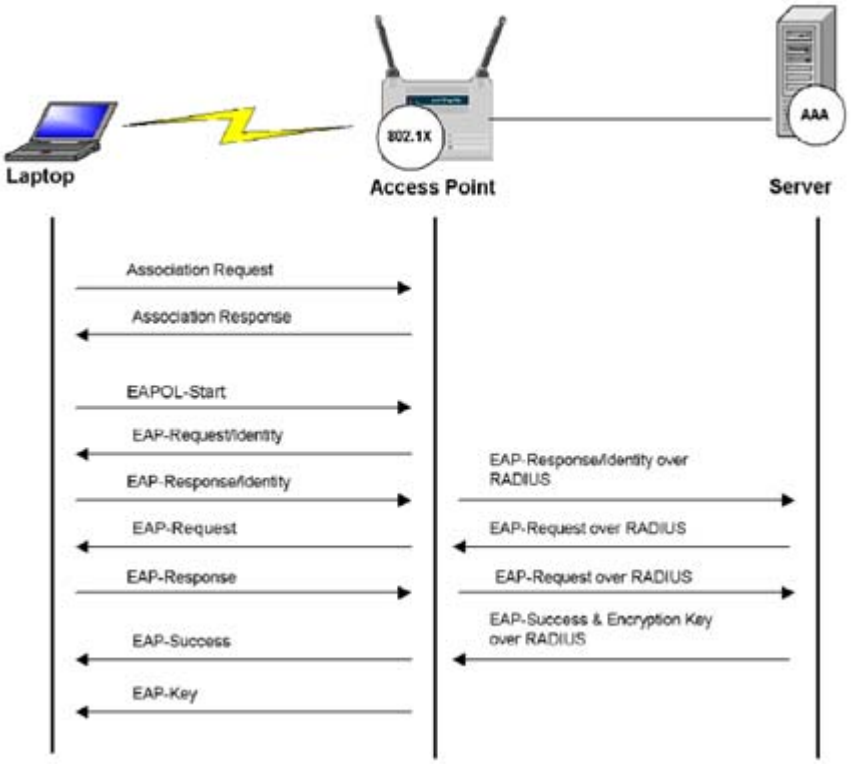

Figure 3 : 802.1X in WLAN

Some of the advantages of using 802.1X in WLANs are:

- Control at the network edge, avoiding the access of unauthenticated intruders.
- Dynamic Session Key Management, improving security.
- Low overhead.
- Utilizes open standards, and it can be integrated with others standards required for authentication, authorization and accounting as for example RADIUS.

3.2 RADIUS Protocol

As mentioned before the authentication server is usually implemented with the Remote Authentication Dial In User Service protocol aka RADIUS, which is a standard protocol used to provide authentication, authorization and accounting (AAA) services. RADIUS has client/server architecture and in RADIUS context the client is an entity that acts as a client in RADIUS messaging and it is called Network Access Server (NAS). Then the final user or device that is authenticating is not a RADIUS client and they are connected through the NAS.

The process is initiated by a RADIUS client or NAS who sends users credentials and connection parameters information using RADIUS messages to a RADIUS server and waits for the answer of the server. The RADIUS server contains a database with user information and performs the authentication and authorization of the RADIUS client request, and sends back a RADIUS message response. RADIUS clients also send

RADIUS accounting messages to RADIUS servers. The authentication in the server can be done using different algorithms. Officially RADIUS uses the ports UDP 1812 for authentication and 1813 for accounting, some old specifications use ports 1645 and 1646.

RADIUS is a simple protocol that consists of just eight messages, where only the first four are in the base specification; the eight RADIUS messages are briefly described in the following list:

- Access Request: message generated by the NAS towards the server on behalf of a user access request.
- Access Challenge: message sent from the RADIUS server to the NAS to question the NAS or the user about some challenge or perform a negotiation.
- Access Accept: message sent from the RADIUS server to the NAS to indicate successful completion of the request.
- Access Reject: message sent from the RADIUS server to the NAS to indicate the rejection of a request.
- Accounting Request: message sent by the NAS to the accounting server to convey accounting information regarding the service provided to the user.
- Accounting Response: message sent by the accounting server to the NAS to acknowledge the accounting information sent by the NAS and to indicate the result of the accounting function.
- Status-Server and Status-Client: experimental messages.

RADIUS can carry information regarding many different functions in the form of attributes. Typically the main body of an Access Request and Access Challenge messages are used to carry attributes from NAS to RADIUS server and vice versa, thus two Access Request messages perform different functions depending on the attribute they carry, as in the case shown in figure 3.

3.2.1 Security of RADIUS

RADIUS provides basically two security functions, one is attribute (usually password) hiding and the other is the authentication of certain messages. Both functions are performed using MD5 hash function and a shared secret between the NAS and the RADIUS server.

The used of this shared secret has caused vulnerabilities in RADIUS deployments such as:

- Static manually configured shared secret: usually the shared secret is configured manually at the NAS and there is no dynamic or automatic method to set or update it in the protocol, in consequence the shared secret never changes in the NAS lifetime and it is commonly replicated in all NAS belonging to the same operator.
- Shared secret lookup: to prevent spoofing, the RADIUS server uses the source IP address in the RADIUS UDP packet (rather than NAS IP address or ID attributes) to look the shared secret up. This is due to the need to support hop-by-hop security when RADIUS proxies are implemented and due to the fact that the NAS

ID is only added as an attribute to the access request payload. This arrangement can potentially cause problems in case where the NAS IP address may change.

- Proxy chaining: in deployments using RADIUS proxies between the NAS and the RADIUS server, the NAS only shares the secret with the first hop AAA proxy and not with the backend server. It means the NAS communicates with the RADIUS server based on a chain of trust rather that a direct trust relationship. One compromised proxy can cause security problems.
- Transport protection: attribute hiding provides selective application layer protection. It does provide any security protection for RADIUS messages or the protocol layers that these messages are riding on. This means the IP address can easily be spoofed or other attributes could be changed.

Then, what to do in order to secure the implementation of RADIUS protocol? Some security solutions that have been used to have a reference are:

- Use RADIUS with EAP protocol.
- Use IPSEC protocol.
- Use long shared secrets.
- Use different shared secrets between a server/client pair inside the network.
- Use the Message-Authenticator attribute in all Access-Request messages.
- Use a cryptographic-quality random number generator to generate the request authenticator.

3.2.2 RADIUS Proxy Operation

In this deployment, the request is not sent directly from the NAS to the RADIUS server; in fact, the original AAA request is generated by the NAS and is sent towards proxy1. Proxy1 examines the request and may forward it to proxy2. Proxy2 will then forward the request to the user's home server, as shown in the figure 4.

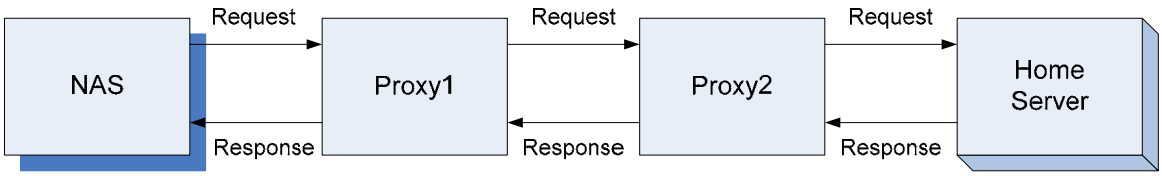

Figure 4: RADIUS Proxy Chain

Both Proxy1 and Proxy2 may find it necessary to modify the attributes in the packet after implementing some local policies. It is even possible that any proxy might simply reject the request and send an AAA reject message to the previous proxy or the NAS. The proxy may even challenge the previous request sender. The proxies are also allowed to hide information in the packets, when needed. These functionalities were provided to compensate for the fact that RADIUS does not support capability negotiations between a NAS and a server that accommodate different feature sets. Usually this configuration is used in roaming environments.

3.2.3 A free Software RADIUS Server Implementation

For practical purposes it is possible to use Freeradius which is an open source radius server that provides AAA functions for different types of network access; it is open source and is widely used in the academic community. The current server version at the time of the present work is 2.0.1. For additional information visit the web page (www.freeradius.org).

Freeradius also provides some tools to facilitate the interaction with the radius server, as for example:

- Radtest: command that generates RADIUS packets to test RADIUS server, it shows the reply.
- Radeapclient: is a radius client program that uses a configuration file with EAP parameters and then performs the authentication on the RADIUS server.

4 SECURITY IN MANETs

The features of MANETs as:

- Self configuration and self management.
- Dynamicity of nodes.
- Lack of any infrastructure.
- Need of routing protocols and nodes acting as routers.

makes difficult to implement wireless basic security mechanism; additionally it offers a new vulnerability that can be exploited by an attacker, the routing protocol.

4.1 Attacks to Wireless Routing Protocols

MANETs are created according to participant needs, the cooperation among nodes is fundamental in the network establishment and routers selection, where trusted relationships among nodes are assumed.

Unfortunately this assumption is not good from the security point of view because it can be exploited by hackers that could hear the communication and attack the routing protocol from inside or even outside the network. In any case they could cause denial of service, inject erroneous routing information, replaying old routing information or distorting routing information causing excessive traffic load, loops in the network, unintended network partitioning, inefficient routing and even the collapse of the network. In consequence it is important to provide solutions to guarantee the security in MANETs, and this is actually a very active research field.

4.2 Security for Wireless Routing Protocols

In [9] they classify the security at the network layer in two categories: secure ad hoc routing protocols and secure packet forwarding protocols. In the first case the objective is to enhance the existing ad hoc routing protocols, where the mobile nodes sign its routing messages using cryptography keys; nodes with no key cannot participate in the routing process.

In the second case the security goal is to ensure that a node only forward packets according to the routing table using a detection technique and a reaction scheme.

In our study we are interested in exploring the possibility of having AAA services in a MANET and use it as part of securing a routing protocol.

4.3 Distributed AAA Services in MANETs

In a recent work [1], there is a proposal that allows the implementation of AAA services in these scenarios in order to help with the deployment of practical or business implementations. The main idea is to have a framework where the AAA services are available in a distributed manner inside the MANET; it means the main AAA service is decomposed in three different subservices that are distributed among several nodes inside the network. The three services are then: Authentication (Aaa), Authorization (aAa) and Accounting (aaA).

This distribution involves new challenges to network managers, because sensitive information will be stored in some nodes inside the MANET, without the usual protection we could have in wired networks. Additionally part of it would be broadcasted, forwarded and routed by any of the mobile nodes. That means we need to create or rethink the use of security mechanisms in order to have trusted relationships among the nodes to protect the network from possible attacks.

But why to create such infrastructure, according to [1]: One obvious and original consequence of the secured framework is the extension of the access coverage by cheap investment using ad hoc technology. Also, it will offer the integration of ad hoc technology in the service value chain by the introduction of a new service provider entrant (ad hoc network service provider), and a new network access provider (ad hoc network). The classical operator then will make profit by offering in addition to his classical services (access to Internet), new services for ad hoc nodes. For instance, it will act as a third party between the servicing ad hoc nodes, and the customers (local ad hoc nodes). This will be to guarantee the AAA service and a secured transaction for exchanged services (peer-to-peer, packet forwarding, resource consumption…). In the next section we will evaluate possible implementations of this proposal.

As explained before, we are interested in practical networks solutions that enable service provider's to use ad hoc technology in the value chain. Now, among the three possible A's distributed services we are particularly interested in the authentication one. According to [1] the topology of the network having distributed AAA services would be as shown in figure 5, and it is called "AdIN" (Ad hoc/Infrastructure).

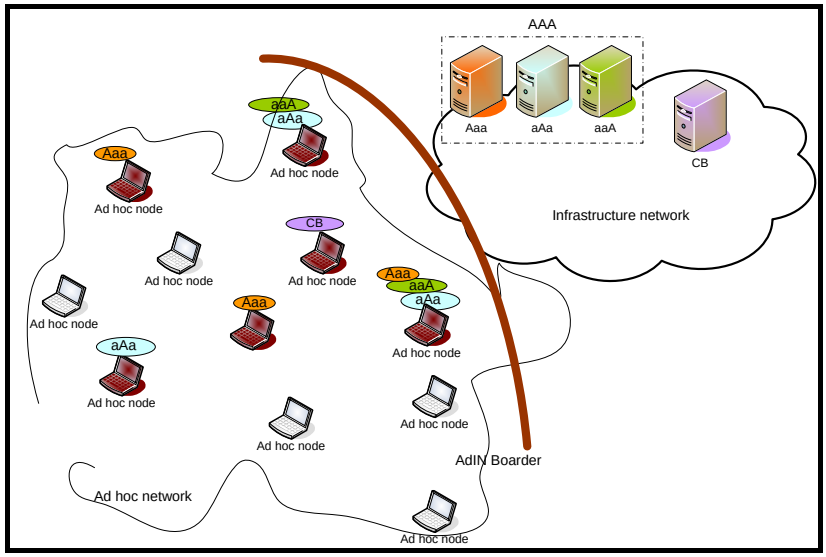

Figure 5: Distributed AAA services Architecture

In figure 5 we can observe that the services Aaa, aAa and aaA are distributed among the different ad hoc nodes, trusted relationship among them is assumed. However, since the Aaa service is the most critical because it authenticates all the nodes entering the MANET, then it could be implemented as an inter-domain service, thus this node that is in

the AdIN border communicates with the infrastructure network for authentication purposes. The CB service refers to charging and billing.

Now, our challenge is to find practical solutions in order to implement the authentication service inside MANETs. In order to answer this question we propose two possible solutions:

- Use virtualization technology
- Or using a secured routing protocol.

In the first case the idea is to build several special configured nodes inside the network that will embed a secure environment thanks to the virtual operating system, this is to provide the authentication (Aaa) service as shown in figure 5. In the second case the purposed solution is to modify a routing protocol in order to control the routing protocol using the authentication service. In other words, the authentication will be implemented as previous step to the routing process. As in the previous case, there is a special node running a RADIUS server, this solution is explained in details in chapter 6.

For our study, we consider the use of RADIUS protocol to provide the authentication service inside the MANET and in the case of real implementations we use Freeradius server since it is open source and we have some previous experience working with it. Both solutions are explained in the next chapters.

5 AUTHENTICATION IN MANETs THROUGH VIRTUALIZATION

Virtualization refers to the use of the same hardware by different operating systems and multiple applications which run independently one of the other and each of them believed they are alone using the hardware, but in fact they are just using virtual resources. The number of OS is limited to the real hardware capabilities. In this environment the different running operating systems are known as guests and they run over a virtual platform created by a host software or control program. The running environments are referred as virtual machines. The difference between virtualization and traditional server models is show in figure 6.

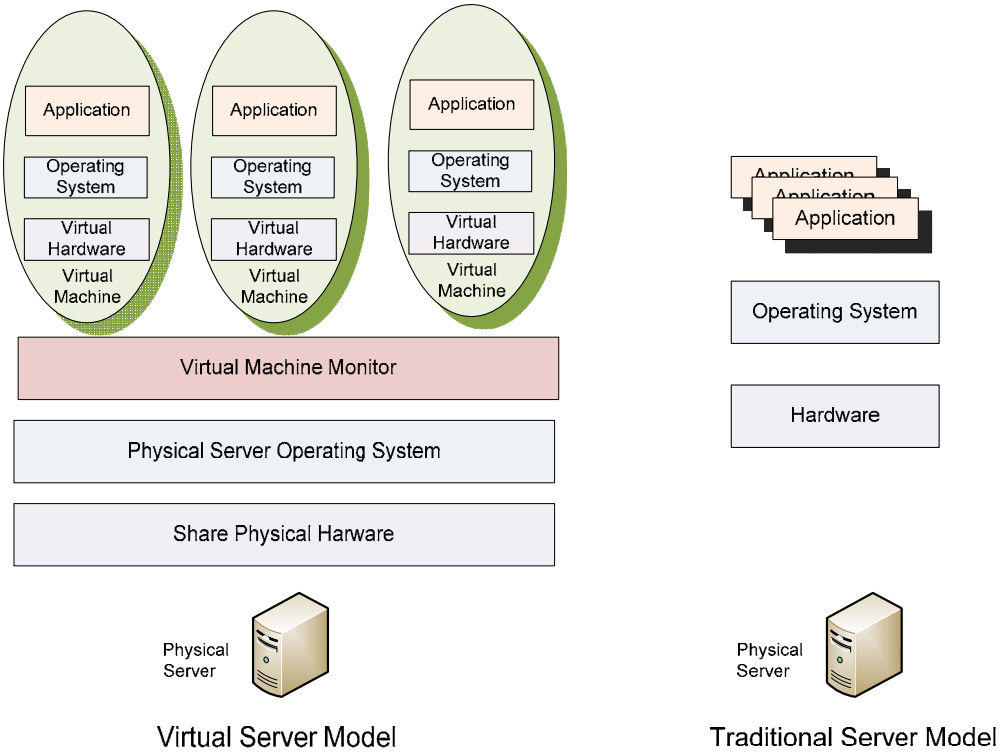

Figure 6: Server Models

There are different techniques to perform virtualization, among them we have:

- Emulation or simulation: in this case all the hardware functions of one system are duplicated in the virtual environment. One advantage of this approach is the use of operating systems without any modification. The access to the real hardware is performed by turn. Examples of virtual packages in this category are: Bosch, DOS Box and GXemul.
- Virtualization: enough hardware is simulated to support an unmodified operating system running in the virtual machine, as the case of VirtualBox, Parallels, QEMU and some VMware packages.
- Para-virtualization: the guest operating system is modified and is aware of the process. The OS then communicates with a hypervisor which has the control of

the hardware; in this case the performances may be closer to native speed. Some virtualization packages that use para-virtualization are: Xen, VMware Workstation, and Sun xVM.

Some of the reasons to use virtualization technologies are:

- Server Consolidation and Infrastructure Optimization: Reducing the number of physical machines required to run many physical servers. Different to the approach one machine, one server.
- Physical Infrastructure Cost Reduction: Reducing the number of required hardware leads to reductions in cost, power and cooling requirements, resulting in lower IT costs.
- Improved Operational Flexibility & Responsiveness: Decreasing time on repetitive tasks such as provisioning, configuration, monitoring and maintenance.
- Increased Application Availability & Improved Business Continuity: Reduce or eliminate interruption time for services. Servers can have hot standby backup.
- Improved Desktop Manageability & Security: Deploy, manage and monitor secure desktop environments that end users can access locally or remotely, with or without a network connection, on almost any standard desktop, laptop or tablet PC.
- Testing and training: Useful in software development, patches evaluations and operating system courses are done over virtual machines without risking the real system.
- The need to run legacy software: Usually operating systems evolve while legacy software cannot be changed.
- Need to run different operating systems at the same time.

5.1 Virtualization Packages

There is a wide offer of virtualization packages which can be chosen according to the required features or individual needs, for example, host operating system, processor architecture and price could be parameters to consider for the choice. We only consider free software solutions and among them we study VirtualBox and Xen.

5.1.1 VirtualBox

From VirtualBox manual we have: “innotek VirtualBox is a family of virtual machine products targeting desktop computers, enterprise servers and embedded systems. Due to its modular architecture, VirtualBox can be deployed in any environment where x86 systems are to be virtualized on x86 systems. (With “x86”, we are referring to 32-bit CPUs from AMD and Intel as well as compatible CPUs from other vendors, plus 64-bit CPUs in 32-bit mode.)”. Current VirtualBox version at the time is 1.5.

5.1.1.1 VirtualBox Practical Experience

VirtualBox is an easy package to install and work with. We installed it under Linux Fedora Core 8; then a Windows XP virtual machine (VM) was created using VirtualBox graphical interface and following the manual instructions. The graphical interface is intuitive and allows the management of all the VMs and their resources. At the beginning, when you want to work with the VM you click on it and a key capture feature is activated, in order to go back to the host system a special key combination has to be pressed. To optimize and improve the performance and usability of the VMs it is possible to install some additional software from VirtualBox called “guest additions”, which consist of device drivers and system applications. For example, after the installation of the additions it is possible to work on the VM as it is any other application; also they provide better video support.

Once installed, a VM can have up to four virtual PCI Ethernet cards; the default network mode is network address translation (NAT), however, we changed it to bridge mode to have the VM in the same network as the guest system. The network configuration of the VMs works with the wired interface of the host system, thus in order to make it work with the wireless interface, it is necessary to create another bridge, use proxy ARP between the VM and the wireless interface and also add a static route in the kernel that points to VM.

Finally, some basic internetworking tests were done to observe the behavior of the windows XP guest and learn about its interaction with the Linux Fedora host, having in mind a comparison with another virtualization system, which will be useful later in our work.

Another issue we have to mention is related to the update of the Linux Fedora core. Every time the Fedora core was updated, the VirtualBox application did not work, thus it was necessary to use the command line application to recompile VirtualBox and be able to use it again, fortunately the guest VM is not affected in the process.

5.1.2 Xen

The Xen hypervisor, the powerful open source industry standard for virtualization, offers a powerful, efficient, and secure feature set for virtualization of x86, x86_64, IA64, PowerPC, and other CPU architectures. It supports a wide range of guest operating systems including Windows, Linux, Solaris, and various versions of the BSD operating systems.

Xen also supports the new generation of AMD Pacifica and Intel VT-x chipsets and can run an OS on these chips without any modifications by using a version of the hypervisor called the Hardware Virtual Machine (HVM). HVM mediates between the guest operating system and the hardware and passes on the calls made by the guest to the physical hardware. So you can run Microsoft Windows on these chips using Xen.

Xen is an open-source para-virtualizing virtual machine monitor (VMM), or “hypervisor”, that can securely execute multiple virtual machines on a single physical system with close-to-native performance. Xen facilitates enterprise-grade functionality, including:

- Virtual machines with performance close to native hardware.
- Live migration of running virtual machines between physical hosts.
- Up to 32 virtual CPUs per guest virtual machine, with VCPU hotplug.
- x86/32, x86/32 with PAE, and x86/64 platform support.
- Intel Virtualization Technology (VT-x) for unmodified guest operating systems (including Microsoft Windows).
- Excellent hardware support (supports almost all Linux device drivers).

5.1.2.1 Xen Practical Experience

Xen is more complex than VirtualBox system and requires good hardware to be installed. For its installation we tried first to compile the Xen kernel according to Xen book instructions without success. Then we found that it can be installed using Fedora packet manager in the graphical interface or yum application in command line interface. The current version of Xen under Fedora core 8 for the time of this work is 2.6.21.7-3.fc8xen.

When Xen is installed it can be activated using Fedora Virtual Manager graphical interface, which is as intuitive as the case of VirtualBox. The default installation mode in Xen is paravirtualized, to enable unmodified guest installations it is necessary to have a processor with the option of virtualization technology enabled. It means you have to enter the BIOS setup of your computer and activate this option, which was disabled from factory, After this process Xen is ready to create VM with unmodified guest operating system.

We installed a Windows XP guest and again, the default network mode is NAT, but we did not find the option to change it to bridge mode. So, in order to make Xen work in bridge mode we used the command line utilities of Xen to create the VM again. In this case it was necessary to create a configuration file for the VM, and when it is created it can only be managed by command line interface because the Virtual Manager does not recognize it. The device sharing is complicated and we did not try it.

Also to make the VM work with the wireless interface, we changed the bridge configuration and we used proxy ARP between the VM and the wireless interface and also we added a static route in the kernel that points to VM. With Xen we did not have guest additions and the VM works in capture mode, thus you need to release it before going back to the host system. Finally we observed the behavior of the VM and its interaction with the host.

5.2 Security in Virtualization

As virtualization mainly implies the use of different operating systems in only one physical device, a computer or server, we have to consider the possible sources of security problems due to the known vulnerabilities of the OS used and also the ones related to the own virtualization system which is also a program and even the security of the physical computer and applications to be used inside the virtual machines.

Previously we mentioned that virtualization is used in order to have some advantages over traditional dedicated servers, for example: live migration, load balancing, patch testing, better use of hardware, etc. Thus the use of different operating systems increases the security challenges.

In virtualization one of the most important concepts is isolation, it means, the different VMs cannot interfere with another or with the host system. A failure or crash of one machine shall not affect other guest or the host environment.

The security risks of using virtualization are:

- Technologies that share data between the VM and the host. A compromised VM can open a door to attack other VMs or the host system. Also exiting operating system security threats need to be considered here
- The possibility of logging keystrokes and screen updates passed across virtual terminals in the VM
- Some VMs that do not focus on isolation or bugs that may result in isolation compromise
- VM monitoring from another VM without being configured
- DoS against another VM
- External modification of VM
- Unauthorized modification of hypervisor. The hypervisor operates like an operating system and could require patching. If a hypervisor needed to be patched all virtual machines would have to be brought down
- Detecting and evading virtualization
- Trapping malicious code via virtualization

As we can observe secure VMs is rather complex work and according to [3] virtual machines are more difficult to secure, especially with today's static security devices. The traditional firewall DMZ to separate servers is not flexible to be used with VM. It means we need new mechanisms to deploy security. One solution is to consider the security outside the VM and within or below the hypervisor, for example, implementing a soft firewall to control the access between the physical network device and the virtual network interface of the hypervisor. Also verify available tools for the VMs.

Another fundamental security measure is to ensure that privilege escalation is not possible within the VM that could compromise the host system.

In [4] the threats are classified according to the level of compromise of the VM monitor and the host manager access to the VM. They performed security audits of different virtual machines implementations and they found that almost all of them have exploitable security flaws.

Another possibility of attack is evaluated in [5] where they demonstrated that the presence of virtual machines can be detected and this could lead to the use of malicious attacks. In [6] they developed a malware to attack VMM and gain control over the host system and in [7] an empirical method shows how to attack the live migration functionality of two popular VMM.

Then let’s see what the recommendations to safely deploying virtualization are:

- Treat Virtual Machines like services that can be compromised; use chroot, systrace, acls, least privileged users, etc.
- Limit VM resources and external services you don't use (DHCP daemons, etc.).
- Maintain the integrity of guest operating systems, protect the kernel using standard procedures of disabling modules, /dev/mem, /dev/port, etc.
- Keep guest software updated with published vulnerabilities.
- Keep VM software updated.
- Avoid guest that do not operate in protected mode.
- Digitally signing of VM and validating the signature prior to execution.
- Limit physical access to host. Access to the hardware of the virtual infrastructure can cause similar attacks on non-virtual hardware.
- Secure the host operating system: limit users accounts, network services, installed programs and applications. Keep it updated.
- Use encryption for secure communication from guest to host or from management devices to host.
- Consider security rules that are applied to normal servers to be applied to VM
- Avoid file sharing among VM.

5.3 Authentication Service Using Virtualization in MANETs

In this section we consider from a theoretical point of view, the use of the virtualization technologies studied before in order to have a solution to deploy secure services in MANETs by using special ad hoc nodes. These nodes will run at least two environments, one for authentication controlled by the operator and one for the user. Virtualization with isolation concept will securely separate the two environments embedded in the called trusted ad hoc node.

The main idea is to have a distributed authentication service and bring it as close to the users in a MANET (cf. figure 7). To do that, an operator should configure several nodes that will be part of the MANET; these nodes will have two different environments using virtualization software. The number of these special nodes will depend on the network size and the desired coverage service.

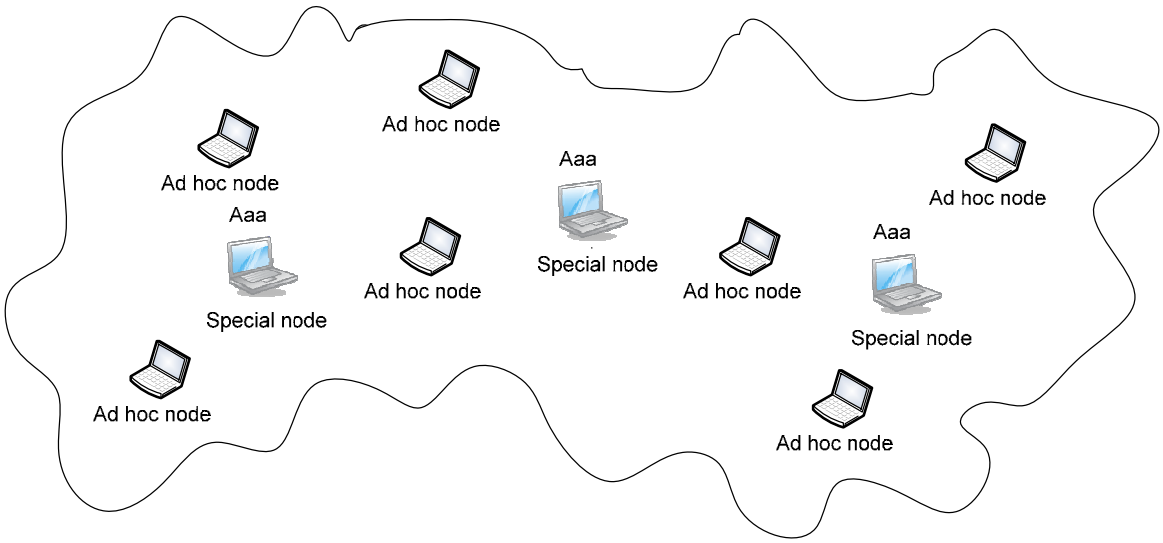

Figure 7: MANET with Virtual Machines

In the native environment of the node, the operator installs a RADIUS server with a database of the customers that could use the services offered by the operator itself or a third party. The service offered to the customers is loaded in a virtual machine.

Under these conditions, there should be a trust relation between the operator and the service provider if it is a third party, since sensitive data will be stored in the host operating system and the service provider will have physical access to the node.

Additionally it is important to consider the security issues regarding virtual machines implementation. However, the advantage of having virtual environment for the service offered is the possibility to migrate the service to another virtual machine in the case the service is down or is attacked by hackers reducing the down times and increasing service availability.

Thus, the proposed solution would be implemented as follow:

- Prepare several laptops with Linux as host operating system.
- Install virtualization technology.
- Install Freeradius server, and customers' database. For security database content should be encrypted.
- Configure RADIUS server and policies.
- Create virtual machines for the service provider according to the required operating system.
- Prepare the booting sequence of the laptops to leave them ready to work with all the services activated.

Next step should be signing a confidentiality agreement with the service provider and deliver the special ad hoc nodes to users who would like to provide the connectivity via

the ad hoc service to other customers. The incoming customer will buy the service and use their accounts with the telecom operator to access it.

5.3.1 Authentication Time with Virtual Machines

In order to have an idea about the authentication time, we built a scenario with three nodes, one of them with a virtualization application, as shown in the figure 8:

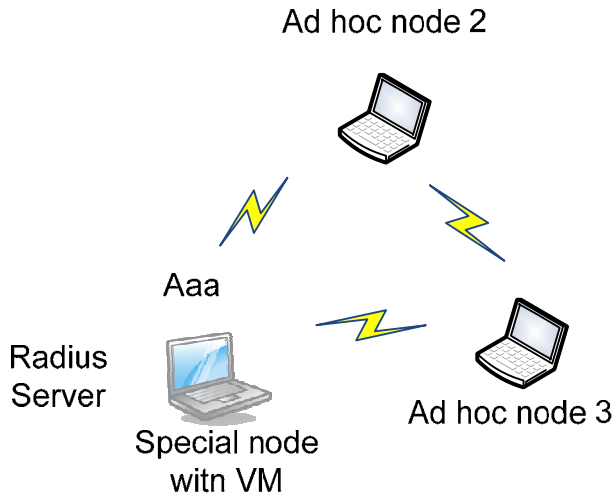

Figure 8: MANET Topology for Measurements

All the nodes are using Linux as operating system; the virtual machine is Windows XP and was created using Xen and VirtualBox as it was explained in previous sections, the nodes has no mobility. The equipment used for the test is listed in the following table:

Special node with VM	Ad hoc node 2	Ad doc node 3
HP xw4600	Dell Latitude D410	Dell Inspiron 6400
Core2 E6850 @ 3GHz	Pentium M @ 2GHz	Core2 T7200 @ 2GHz
4GB RAM	2GB RAM	2GB RAM
WPN111 Wireless USB Adapter	Intel PRO Wireless 2915 802.11 a/b/g Wi-Fi	Intel PRO Wireless 3945 802.11 a/b/g Wi-Fi
250GB HDD	40GB HDD	160GB HDD
Fedora Core 8 + ndiswrapper + Freeradius	Fedora Core 8 + radtest	Fedora Core 8 + radtest

Table 1: Equipment list

From table 1 we can see that node 1 has no internal Wi-Fi interface, so an USB Wi-Fi interface was added using “Ndiswrapper”; which is a free software driver wrapper that enables the use of Microsoft Windows drivers for wireless network devices on Unix-like operating system. This is due to the fact that there are no native Linux drivers for many

wireless network adapters. For more information visit their web page (ndiswrapper.sourceforge.net).

The special node in figure 8 has Freeradius server and the virtual environment; the nodes are using the radtest command to perform the authentication and the time is measured with Wireshark.

Two scenarios are considered per virtualization application, the first the VM is ON but with no activity, in the second the VM is busy with a file transfer with the opposite node, it means, if we measure the time for node 2, the window VM is transferring a file with node 3 and vice versa. The transfer was done using Filezilla application.

VM type in Special Node	Node 2	Node 3
VirtualBox VM	0,002881	0,001150
VirtualBox VM busy	0,010312	0,010742
Xen VM	0,003245	0,001880
Xen VM busy	0,010656	0,027810

Table 2: Authentication Times expressed in Seconds

We can observe in the results how the authentication times increases when the VM is busy, which is reasonable since the kernel of the special node has more work to do, forward packets to/from Windows VM to one of the nodes; additionally the occupation of the air interface also increases. Then, it is important to evaluate all these factors in order to dimension the topology of the network and guarantee a target performance.

5.3.2 Difficulties

One of the main difficulties during this phase is the selection of the virtualization application because there is a wide variety; the selection depends on factors like: hardware, host operating system, guest operating system, security between VM, price and available documentation for example. After the selection you need to face the installation procedure which can be easy as the case of VirtualBox or as complex as Xen.

Also we had problems with Ndiswrapper application when the Linux kernel is updated, Ndiswrapper was disabled and could not be compiled again, and thus it was necessary to search for solutions in internet to solve the problem, hopefully we found it (doc.fedora-fr.org/wiki/Wifi-Installation_de_Ndiswrapper).

In the case of Xen the problems were related to the wireless interface. For example, Xen was installed in node 2 and the wireless interface card was not recognized, even when the card was present and active under the normal Linux core. According to some post in Internet, the wireless card has to be added using Ndiswrapper and the windows driver, procedure that we did not apply. However we managed to install Ndiswrapper for node 1 after we follow the instructions given above but we had to install an additional component because is Xen core.

And finally, virtual machines are mainly focus on working or wired Ethernet interfaces, to make them work with Wi-Fi interface the procedure is different to the one given in the virtualization manuals.

5.3.3 Work Improvement

This proposal could be improved and exploited in depth, for example it could be interesting to find a minimal configuration for the host operating system that consumes few resources and offers protection to the authentication information, then develop a booting sequence where only the virtual machine environment can be seen, while all the work done on the host OS is hidden.

6 AUTHENTICATION IN MANETs WITHIN THE ROUTING PROTOCOL

As we studied in chapter 2 secure wireless routing protocol in MANETs is important in order to avoid possible attacks. The MANET we are interested in has a limited number of users and reduce mobility, thus we will select OLSR as our routing protocol.

6.1 Optimized Link State Routing Protocol - OLSR

OLSR is defined in the RFC3626 and available as open source software in internet at www.olsr.org. OLSR is a proactive routing protocol that is implemented as a daemon "olsrd", the current version for the time of this work is 0.5.

The default behavior of olsrd is different from the RFC definition because it is more adapted to the reality of wireless environments, olsrd considers link quality measurements in order to determined the routes to destinations and not only the best path as described as standard behavior. However this feature can be disabled in the configuration file and then it is RFC compliant.

Olsrd is flexible and also offers the possibility of adding features or modify the global behavior through the implementation of modules that are called plugins, as for example, secure OLSR.

6.1.1 Securing the OLSR Protocol

Secure OLSR is a solution which adds a security mechanism to the protocol behavior. sOLSR uses a cryptographic shared key to sign all the packets in order to ensure the integrity of OLSR control traffic data; only the nodes that share the key can participate in the routing domain. Messages without verifiable signs are discarded.

Additionally, to prevent replay attacks, secure OLSR uses timestamps, so the nodes exchange their timestamps before allowing the flow of any traffic between them. In this exchange process, three new messages types are introduced:

- Challenge message
- Challenge-response message
- Response-response message

However, the exchange occurs between neighbors that have not registered timestamps of each other and where the traffic cannot be validated by the signature check. It means these messages are signed internally, and they carry their own digest and they are never stacked with other OLSR-messages but rather sent in OLSR-packets of their own.

6.1.1.1 The Timestamp Exchange Process

When A receives a signed message from a neighbor B, for which A has no registered time value, A initiates the timestamp exchange process. It means A sends a challenge message. The message is broadcasted since A might not have a route to B. The challenge message contains a 32-bit nonce. A then signs this message with a digest of the entire message and the shared key.

B has to respond to this message with a challenge-response message. B first generates the digest of its IP address, the received nonce and the shared key. B then generates a 32-bit nonce and transmits the nonce, the timestamp of B, the digest and a digest of the entire message and the shared key.

When A receives the challenge-response message from B, it first tries to validate the data. If the digest can be validated, then the timestamp of B is used to create the difference of time between A and B. A then generates a response-response message and broadcasts it to B. The message contains the timestamp of A, a digest of As address, the nonce received from B and the shared key and a digest of the entire message and the key.

Finally, when B receives the response-response message from A, it tries to verify the digests. If they can be verified, B uses the received timestamp to register its time difference to A and the process is now completed.

6.1.1.2 Secure OLSR Implementation

The secure version of OLSR is implemented as a plugin of the olsrd daemon. It captures all the incoming traffic, verifies the packets and removes the signature message and updates the size field of the OLSR packet header. The plugin also intercepts all outgoing OLSR traffic to add the signature messages and updates the packets size as shown in the figure 9 taken from [8]:

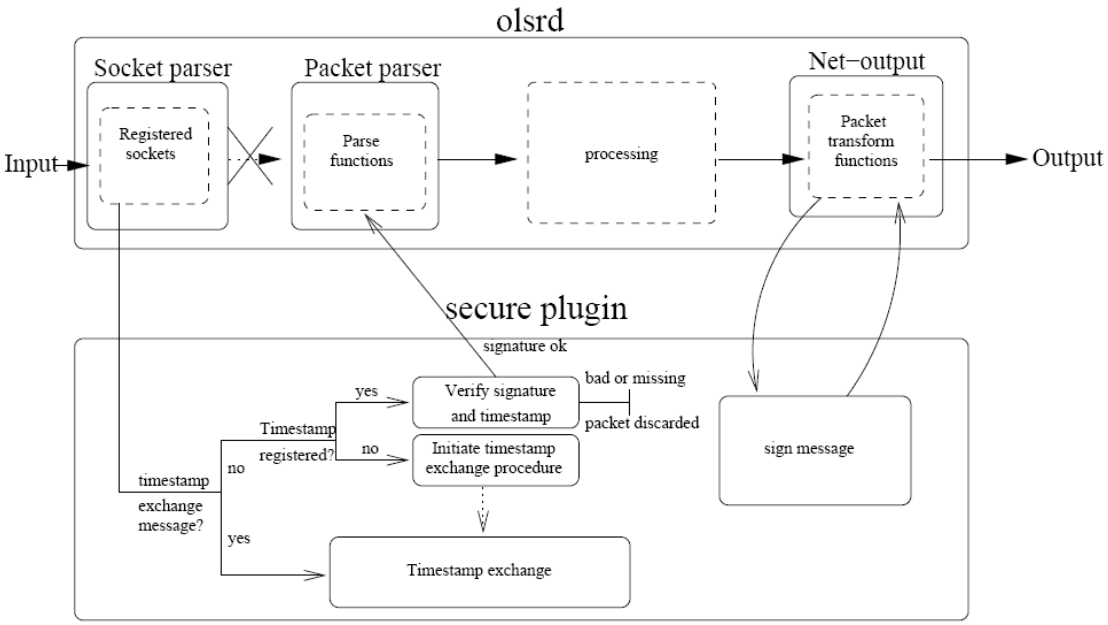

Figure 9: Design of olsrd secure plugin

6.2 Combining OLSR and Authentication in MANETs

Continuing with the purpose of having distributed authentication in MANETs now we propose a second solution. In this case we want to authenticate all the nodes that will participate in the MANET routing domain, if authentication fails the node is not added and the ad hoc routing is blocked.

It means, there will be one special node inside the network which also runs a RADIUS server, which will restrict the access to the routing domain through nodes authentication.

The solution is implemented as a plugin of the OLSR daemon, and it is done modifying the existing secure plugin of olsrd.

6.2.1 Authenticated OLSR Protocol

Authenticated Optimized Link State Routing protocol or AOLSR is a new implementation of OLSR protocol where nodes are required to be authenticated by an AAA server as a previous requirement to participate in the routing domain. It is noted that if AOLSR is activated, OLSR and sOLSR packets are dropped by AOLSR nodes.

This new plugin has been by us developed based on secure OLSR plugin and then it supposes there is an existing key shared among all nodes. The idea of this new version is to add an authentication step as a mechanism to avoid that an attacker that compromised the key could join the routing domain. It means, in the case a node gets the shared key; it won't join the network unless it is authenticated by the radius server.

AOLSR uses the same process of secure OLSR, and it just adds some extra conditions in the behavior, as can be seen in the figure 10.

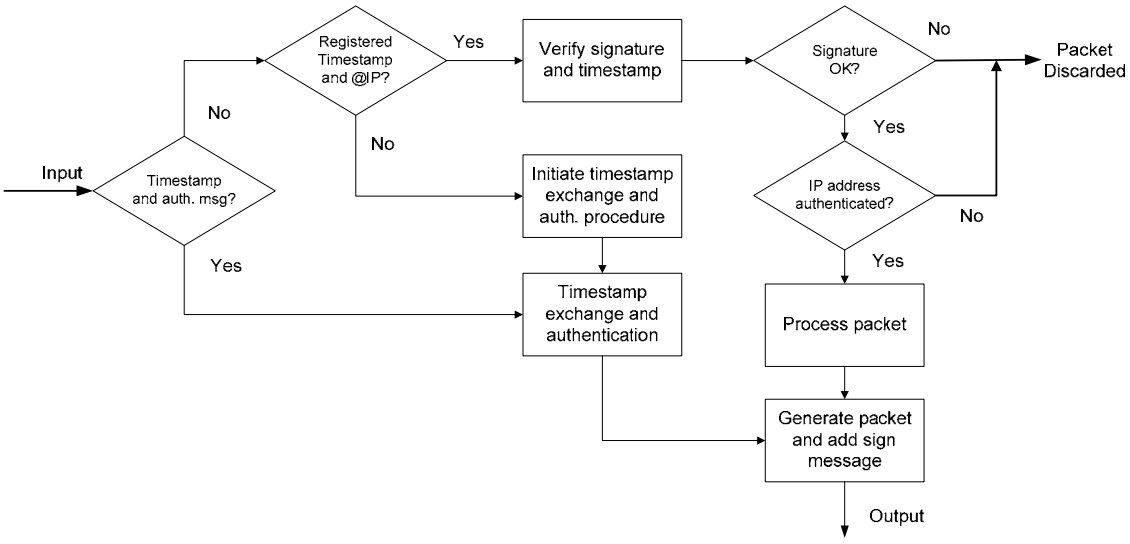

Figure 10: AOLSR - Flow Diagram

Assumptions to develop the application:

- Three nodes: A, B and C are considered in a two hop topology.
- A is a special node that has a radius server for authentication purposes.
- The IP address of the node is used as user name for authentication purposes.
- The radius server is implemented using Freeradius server.
- The radius client request is done using the command “radtest”.

Now, we will describe how the AOLSR protocol functions considering two steps. At the beginning node A is available and node B is the first node coming to join the network. Once A and B are in the same MANET then C will join them.

6.2.1.1 First Node Authentication Scenario

The situation for this case is shown in the figure 11. A is the main node of the network and is running AOLSR and the radius server.

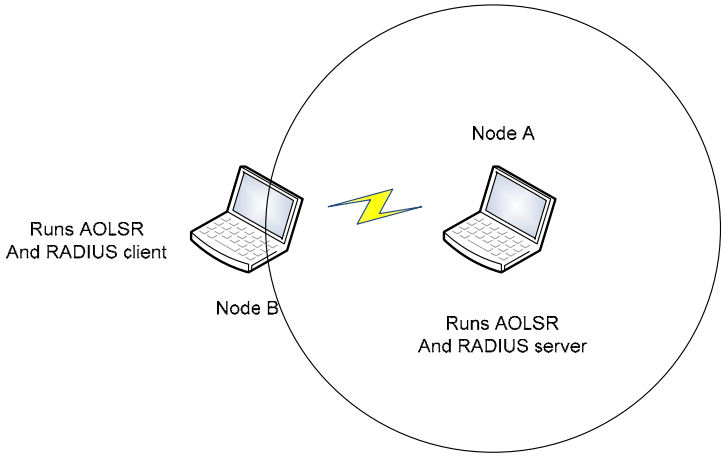

Figure 11: AOLSR - First node authentication scenario

Then B comes running AOLSR and wants to join A. As depicted in figure 12, when A receives any signed message from the new neighbor B, for which A has no registered time value, A initiates the timestamp exchange process. A sends the challenge message. The message is broadcasted since A might not have a route to B.

B has to respond to this message with a challenge-response message, but before that B authenticates with the radius server in A.

When A receives the challenge-response message from B, it first tries to validate the data. If the digests can be validated, then the timestamp of B is used to create the difference of time between A and B. Additionally, A checks the radius log file to verify the successful authentication of B. If it is good, A adds B to its authorized IP address list. Later A generates a response-response message and broadcasts it to B. The message contains the timestamp of A.

Finally, when B receives the response-response message from A, B uses the received timestamp to register its time difference to A and also B adds A in the trusted IP address list.

When A received another message from B, A already has B in the timestamp database, and then A will check also the authenticated IP address list, where B will be if the authentication process was successful. If not, then any packet coming from B is going to be rejected.

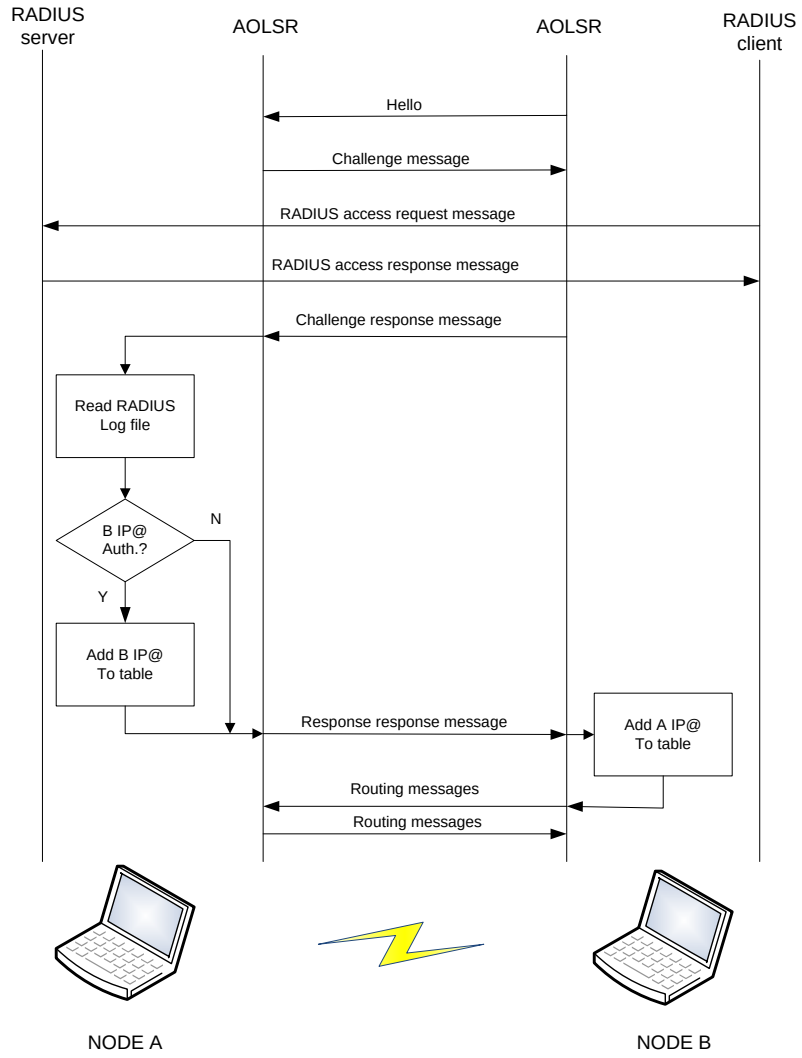

Figure 12: First node authentication - Logical Operation

6.2.1.2 Second Node Authentication Scenario

In this scenario a node C is the new node that wants to join the network formed by nodes A and B. The location of node C is next to B but cannot communicate directly to node A as shown in figure 13.

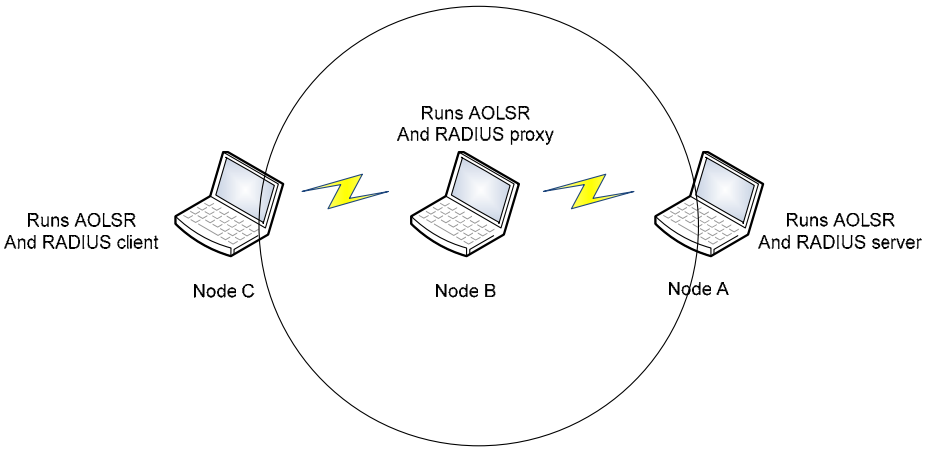

Figure 13: AOLSR - Second node authentication scenario

Since C cannot communicate with A directly, we assume that B is running a radius proxy also which will be used for C authentication.

When it arrives, as shown in figure 14, node C sends a signed message to B, then a process similar to the one described above is initiated.

Node B has no registered time value for node C, thus B initiates the timestamp exchange process sending the challenge message.

When C receives the challenge message, it performs the authentication with the radius proxy running in node B, which forwards the request to the RADIUS server running in node A. After that C sends B a challenge-response message.

When B receives the challenge-response message from C, the timestamp of C is used to create the difference of time between them. Additionally, B checks the radius log file to verify the successful authentication of C. If it is good, B adds C to its authorized IP address list. After, node C generates a response-response message for B which contains the timestamp of C.

Finally, when C receives the response-response message from B, C uses the received timestamp to register its time difference with B and also registers B in the trusted IP address list.

When B received another message from C, B already has A in the timestamp database, and then B will check also the authenticated IP address list, where C will be if the authentication process was successful. If not, then any packet coming from C is going to be rejected.

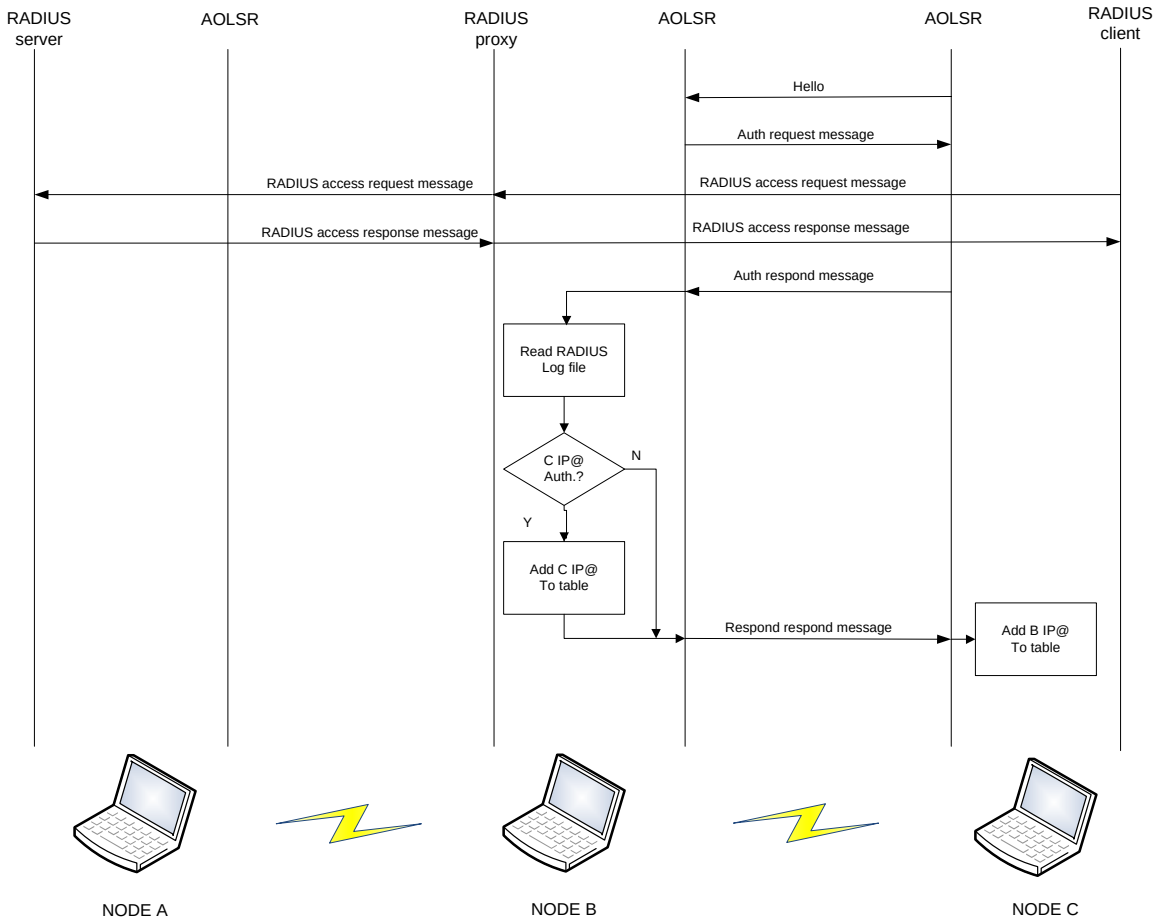

Figure 14: Second node authentication - Logical Operation

6.2.2 AOLSR Practical Implementation

For our practical implementation we add some new functions to the secure plugin code, especially to module olsrd_secure.c. We kept the main functions of the plugin because as explained before it intercepts all the incoming traffic and checks if they are timestamp exchange messages, if yes then the plugin processes the packet according to the packet type, if not it just removes the signature and sends it to the general packet parser function; and this behavior is helpful for our authentication process.

Then, new functions were added to create, add and consult a list of known IP address, and it is done after a successful login of the incoming new node. The authentication is verified reading the RADIUS log file and verifying the authentication of the new node's IP address. Additionally new lines were added to force the look up in the registered IP address list before routing.

The authentication process was embedded in the timestamp exchange process, it means when a new node arrives, it will initiate the timestamp exchange process and will be authenticated at the same time. Now, it was necessary to add a flag to differentiate the node with the RADIUS server and node with the proxy RADIUS in order to avoid that an incoming node tries to authenticate the server.

For the RADIUS server and proxy we use Freeradius server, and for the client the tool radtest.

6.2.3 Difficulties

The main difficulty of this work was related to the use of two different daemons that are independent, olsrd and radius.

For example, when a new node is authenticated, the supplicant node could know if the authentication was successful or not, however, it has to be verified by the node that is acting as proxy or radius server. Thus this verification is done reading the radius log file and searching the corresponding event related to the IP address authentication. Remember that IP addresses are used as user names since in the radius protocol there is no registration of the user's IP address when it requests access, instead, is the IP address of the radius client or proxy that is used in the process.

Also, all the participant nodes must run a radius proxy, which has to be configured according with the network required parameters. As a consequence there should be an authentication service planning in order to establish the parameters that are required to be configured in each node to implement such service.

6.2.4 AOLSR Work Improvements

As it was mentioned before, this work was done adding extra features to the existing secure OLSR protocol, thus it is considered the previous availability of shared keys in all the nodes as a previous condition, and this could be a limitation in some environments. In fact the authentication phase is embedded in the timestamp exchange process, then a first improvement could be a simplification, it means; to remove the timestamp exchange components and leave only the authentication process eliminating the need of cryptographic key.

Another one could be the integration of AOLSR and the timestamps:

- The authentication is combined with the timestamp process; thus we would register the time when a node is authenticated and use it to validate its permanence in the network. The condition to check would be only the validated IP address. It means all the nodes have to re-authenticate periodically and in a time period that is shorter than authentication timeout.

In any case we have to consider an important issue, the authentication is being verified against a radius log file, so if there is re-authentication there would be more data in the file that has to be processed more frequently also, then this access time or even the possibility that the file cannot be read in a certain moment because the radius daemon is writing on it; have to be evaluated carefully.

Finally, some performance indicators have to be defined in order to evaluate the feasibility of a real implementation of this solution.

7 CONCLUSIONS

WLAN has become a popular technology because it allows users connections without wires in any location where it is available. However, its great advantage is also its main vulnerability; the use of the air interface can lead to attacks since the communication can be listened and exploited by attackers in the vicinity of the network. So, security is an important issue to consider in the deployment of WLANs.

The implementation of the security in WLAN depends on the network configuration either infrastructure or ad hoc. In the case of infrastructure there is at least one network node that is connected to a wired network, so the existing wired network security can be used in the WLAN, as for example the norm 802.1X. On the other side, in ad hoc environment there is no infrastructure to count on and any service has to be provided by any of the member nodes. If additionally the ad hoc nodes have mobility then we have a mobile ad hoc network aka MANET. In MANETs it is necessary to have routing protocols which are also a possible target for attacks.

The performance of any solution to provide security in wireless networks has to be evaluated in order to determine how it is affected because security involves some network overhead on a limited resource, the bandwidth on the air.

There are different solutions to provide security in MANETs; among them we were interested in a new framework which proposes to deploy AAA services inside the network to implement security. The main idea is to deploy AAA services in a distributed manner; in other words, the service is separated in three subservices, Authentication (Aaa), Authorization (aAa) and Accounting (aaA) which are offered by some nodes inside the MANET. From these three possible services we focused on finding practical solutions to implement only one of them, the Authentication (Aaa) service.

We proposed two possible solutions to implement the authentication service in a MANET: using virtualization or securing the routing protocol. Virtualization refers to the use of the same hardware by different operating systems and multiple applications which run independently one of the other and each of them believed they are alone using the hardware, but in fact they are just using virtual resources. It means virtualization is helpful to build special nodes in the network that provides the Authentication service on an ad hoc node in a secured manner. While in the second solution, the routing protocol is modified to perform authentication as previous step of joining the MANET routing domain. This is to allow only authenticated nodes to benefit from the routing protocol.

In virtualization there is a wide variety of packages that use different virtualization techniques: emulation, virtualization and para-virtualization. Among them we selected two open source packages to perform a simple comparison, VirtualBox and Xen. VirtualBox is a virtualization application that is easier and simpler to use than Xen hypervisor when you want to create a VM for an unmodified system like Windows XP. In the case of VirtualBox it could be done using the graphical interface, while with Xen it was necessary to enable the virtualization technology in the host computer microprocessor and install the VM using

command line interface. From a performance point of view, we got some authentication time measurements that show better response in the case of VirtualBox, however to conclude that VirtualBox is really better it is necessary to perform a more extensive test, we just wanted to have a reference for the authentication times.

Before using virtualization it is important also to know the possible vulnerabilities of the package to use in order to implement basic security mechanism that can ensure the isolation and protection of the host environment and guest machines.

In the case of securing the routing protocol we worked with an active routing protocol called OLSR. We used an open source daemon implementation (www.olsr.org). This daemon already has a secure plugin which uses a shared key to sign the packets and uses timestamps to avoid replay attacks and establish neighbor trust relationships. Regarding the key, it assumes that it is already available for the nodes that join the network. In our solution we modified the secure plugin of OLSR daemon to add an authentication phase and we called it Authenticated OLSR (AOLSR), thus we assume also the availability of a shared key while the authentication process could protect the network from attacker that were able to get the shared key. It means, there is an initial node that acts as Authentication server and runs AOLSR which will enable the formation of a MANET, then any node that wants to join the network has to run AOLSR and has to be authenticated by the server, if succeeds it will be part of the network. For practical implementation we add new functions in the secure plugin to have authentication, in fact we keep the timestamp procedure and we added access control conditions for authentication and a new table to keep the known and authenticated IP address of the neighbors. We tested the implementation among three nodes and it worked, however a deep evaluation is required to test the performance of this solution.

Finally we can say that we established the bases of two possible solutions to implement distributed AAA services in MANETs, however both of them need to be studied in depth to find an optimized solution and evaluated the performance and security risks considering the difficulties and improvements suggested for both.

8 BIBLIOGRAPHY

Chaganti, P. Xen Virtualization. A Practical Handbook. Packt Publishing. 2007.

Chandra P. Bulletproof Wireless Security. Newnes. 2005

Chaouchi H. and Laurent-Maknavicius M. La sécurité dans les réseaux sans fil et mobiles 2. Lavoisier. 2007.

Hassell J. RADIUS. O'Reilly. 2002.

Innotek. VirtualBox - User Manual. Version 1.5.6. http://www.virtualbox.org/download/1.5.6/UserManual.pdf

Khakpour A., Chaouchi H., and Laurent-Maknavicius M. WATCHMAN: An Overlay Distributed AAA Architecture for Mobile Ad hoc Networks. ares, pp. 144-152, 2008 Third International Conference on Availability, Reliability and Security, 2008

Nakhjiri M. and Nakhjiri M. AAA and Network Security for Mobile Access. Wiley 2005.

Pujolle G. Sécutité Wi-Fi. Eyrolles

Puzar M., Andersson J., Plagemann T. and Roudier Y. SKiMPy : A simple key management protocol for MANETs in emergency and rescue operations. ESAS 2005 No2, Visegrad, HONGRIE (13/07/2005) 2005, vol. 3813, pp. 14-26, [Note(s): VIII-217 p.,] [Document : 13 p.] (31 ref.) ISBN 3-540-30912-8

Swaminatha T, Elden C. Wireless Security and Privacy. Addison-Wesley. 2003.

Tanenbaum A. Computer Networks. Fourth Edition. Prentice Hall. 2003.

The Center for Internet Security. Virtual Machine Security Guidelines. Version 1.0. 2007. http://www.cisecurity.org/tools2/vm/CIS_VM_Benchmark_v1.0.pdf

Tonnesen A. Unik olsrd plugin implementation HOWTO. Version 0.3 for olsrd 0.4.3 and up. http://www.olsr.org/docs/olsrd-plugin-howto.pdf

Xen Source. How to Install Windows on Xen 3.0. www.xensource.com/files/xen_install_windows.pdf

9 REFERENCES

[1] Chaouchi H. and Laurent-Maknavicius M. SAACCESS: Secured Ad hoc ACCess framework. article invité, International Conference on New Technologies, Mobility and Security NTMS'07, Paris, April-May 2007

[2] Chaouchi H. and Laurent-Maknavicius M. Annex 1 of French patent: Intégration de la technologie ad hoc dans la chaine de valeur des télécommunications.

[3] Antonopoulos A. Securing Virtualized Infrastructure: From Static Security to Virtual Shields. Senior Vice-President & Founding Partner, Nemertes Research. http://hackreport.net/wp-content/uploads/2007/03/nemertes-issue-paper-securing-virtualized-infrastructure.pdf

[4] Ormandy T. An Empirical Study into the Security Exposure to Host of Hostile Virtualized Environments. http://taviso.decsystem.org/virtsec.pdf

[5] Ferrie P. Attacks on Virtual Machine Emulators. SYMANTEC ADVANCED THREAT RESEARCH. http://www.symantec.com/avcenter/reference/Virtual_Machine_Threats.pdf

[6] King S., Chen P., Wan Y., Verbowski C., Wang H. and Lorch J. SubVirt: Implementing malware with virtual machines. In /Proceedings of the 2006 IEEE Symposium on Security and Privacy/ (May 21 - 24, 2006). IEEE Computer Society, Washington, DC, 314-327.

[7] Barham P., Dragovic B., Fraser K., Hand S., Harris T., Ho A., Neugebauer R., Pratt I. and Warfield A. Xen and the Art of Virtualization. SOSP'03, October 19.22, 2003, Bolton Landing, New York, USA.

[8] Tonnesen A. Master Thesis: Implementing and extending the Optimized Link State Routing Protocol. University of Oslo. 2004. http://www.olsr.org/docs/report.pdf

[9] H Yang, H Y. Luo, F Ye, S W. Lu, and L Zhang, "Security in mobile ad hoc networks: Challenges and solutions" (2004). IEEE Wireless Communications. 11 (1), pp. 38-47. Postprint available free at: http://repositories.cdlib.org/postprints/618