*Journal of Information Security*, **2012, 3, 224-230**
http://dx.doi.org/10.4236/jis.2012.33028 Published Online July 2012 (http://www.SciRP.org/journal/jis)

# Enhanced Timestamp Discrepancy to Limit Impact of Replay Attacks in MANETs

**Aziz Baayer[1], Nourddine Enneya[2], Mohammed Elkoutbi[1]**
[1]Laboratory SI2M, ENSIAS, University of Mohammed-V-Souissi, Rabat, Morocco
[2]Laboratory LaRIT, Faculty of Sciences, University of Ibn Tofail, Kenitra, Morocco
Email: azizbaayer@yahoo.com, elkoutbi@ensias.ma, enneya@gmail.com

Received April 20, 2012; revised May 16, 2012; accepted May 28, 2012

## ABSTRACT

Mobile Ad hoc NETworks (MANETs), characterized by the free move of mobile nodes are more vulnerable to the trivial Denial-of-Service (DoS) attacks such as replay attacks. A replay attacker performs this attack at anytime and anywhere in the network by interception and retransmission of the valid signed messages. Consequently, the MANET performance is severally degraded by the overhead produced by the redundant valid messages. In this paper, we propose an enhancement of timestamp discrepancy used to validate a signed message and consequently limiting the impact of a replay attack. Our proposed timestamp concept estimates approximately the time where the message is received and validated by the received node. This estimation is based on the existing parameters defined at the 802.11 MAC layer.

**Keywords:** MANET; Replay Attack; Denial-of-Service (DoS); 802.11 MAC Layer; Network Allocation Vector (NAV); Security Countermeasure

## 1. Introduction

Mobile Ad hoc NETwork (MANET) [1] is consisted of mobile nodes MNs which can be either router or normal nodes, are able to communicate by using wireless network interfaces without the aid of any fixed infrastructure or centralized administration. A MANET is considered as an infrastructure less network because their MNs can dynamically establish routes among themselves to transmit messages temporarily. In a MANET, two given MNs can communicate directly when each one is in the transmission communication range of the other one. Otherwise, those MNs communicate throw intermediate MNs that relay their messages [2]. So, the success of a given communication between the sender and receiver MNs is strongly dependent on the cooperation of the intermediate MNs.

Denial-of-Service (DoS) attacks in MANET can seriously affect the network connectivity and disrupt further the networking functions, such as control and data message delivery. In other words, we can say that DoS attacks are capable to harshly degrade the overall MANET performance [3,4]. Indeed, at the physical layer, the attacker can launch a DoS attack with a wireless Jammer by sending a high power signal to cause an extremely low signal-to-interference ratio at a legitimate receiver MN [5]. At the 802.11 MAC layer [6], a replay attack [2,7,8] can be done by intercepting a valid signed messages of MN (the validation is assured by the timestamp concept) and by retransmitting them later in order to produce a DoS attack. At the network layer, a DoS attacker makes the use of the existing protocols vulnerabilities, that can be classified further into three types: routing disruption, forwarding disruption and resource consumption attacks [4,9,10]. At the application layer, a random DoS attack [11] is to flood a network with a large number of service requests. Since the MNs have a limited transmission range, they expect that their neighbors relay messages to remote receiving MNs. The relayed messages are supposed to be performed by intermediate MNs with a good cooperation as a fundamental assumption of MANETs. This assumption becomes invalid when MNs have tangential or contradicting objectives. To overcome their security problems, MANETs adopt new secure solutions [2]. When the most known attacks can be avoided, replay attacks are still subject of various research works due to their easy technique based on recording and re-sending a valid signed messages in the network. So, to avoid those replay attacks in MANET, a timestamp concept is developed [12-15]. Indeed, the timestamp concept permits to a receiving MN to validate the received signed messages. Consequently, a signed message, injected by a replay attacker, arriving with invalid timestamp discrepancy MUST be dropped.

In a MANET, the fixed value of the timestamp discrepancy $\Delta t$ is pre-negotiated between two communicating MNs [13,14]. In reality, the choice of the threshold $\Delta t$ is large enough and consequent MANET becomes more exposed to a wide range of DoS attacks including replay attacks. In this attack, the objective of the attacker is to resend the intercepted signed messages without exceeding the threshold defined by the timestamp discrepancy in the beginning of a communication. So, to avoid this problem a new timestamp discrepancy is required.

In this paper, we present a new timestamp discrepancy to limit the impact of replay attacks. Our proposed timestamp approach is based on the 802.11 MAC layer parameters and on MN capabilities in term of buffering and CPU processing. Moreover, our proposition of timestamp discrepancy enables MNs to limit and reduce the redundant messages injected by a replay attacker.

The rest of this paper is organized as follows. Section 2 presents a related work that gives an overview on DoS attacks related to the 802.11 MAC Layer. Section 3 presents the 802.11 MAC Layer functions. Section 4 presents our improvement. Section 5 presents simulations and results. The conclusion is given in the last Section 6.

## 2. Related Works

In a MANET, communications between MNs are articulated on the 802.11 MAC layer protocol that is vulnerable to DoS attacks [4,16-20]. In papers [17,20], it was discussed that a DoS attacker can exploit the binary exponential back-off scheme to access the channel. Moreover, in the RTS/CTS attack [21], a malicious MN can send the RTS/CTS frames to spuriously reserve the channel without real data transmissions. In the NAV attack [3], an attacker sets large duration values in RTS or CTS frames to reserve channel for maximum time duration. In paper [16], a misbehaving MN can get better throughput by modifying unilaterally the binary exponential back-off algorithm parameters.

Other DoS attack is replay attack [2,4] where the malicious MN can perform attack by recording old valid messages and by re-sending them. This makes other MNs update their internal data structure with stale information (for example updating routing table with a wrong route). The replay attack is achieved when control messages bear a digest or a digital signature without including a timestamp [3,13]. Indeed, while existing mechanisms provide the guarantee to the receiving MN that the message was received as sent, there is no absolute guarantee that a message is being used as intended. The originated MN and the sent message are authenticated, but nothing else. A message that has been captured or intercepted by a malicious MN and is replayed later. It will still be authenticated properly as long as the encryption keys were not changed and the timestamp discrepancy was still valid. Also, it's relatively hard to avoid replay attacks at the 802.11 MAC layer due to the stochastic nature of the DCF and to the similarities between the effects of DoS attacks and congested traffic conditions. Indeed, paper [16] describes that if legitimate MNs can link sequential transmissions from a malicious MN, statistical models can be used to detect MNs that cheat the DCF by choosing low back-off values in order to gain an advantage in terms of throughput. Also, a malicious MN can be readily identified by a detection technique, in which neighbor MNs calculate the actual transmission time by sensing DATA/ACK frames [21]. Assuming the random back-off values are observable, a receiving MN can carry out a sequential test to analyze the distribution of this random variable [16].

## 3. 802.11 MAC Layer Overview

The 802.11 MAC protocols support two models of operation called Distributed Coordination Function (DCF) and Point Coordination Function (PCF). Whereas DCF does not use a centralized control, PCF needs an access point (AP) to coordinate the activity of nodes in its area and to operate only in infrastructure-based networks. When PCF is an optional feature at different 802.11 implementations, DCF is obligatory.

The DCF is based on the CSMA/CA protocol. Before a node starts to transmit a packet, it senses the channel idle for a duration DIFS plus an additional backoff time. The backoff time is an integer multiple of a basic slot duration $\delta$, where the back-off number is drawn randomly in the range $\left[0, \mathrm{CW}-1\right]$, where CW is called a contention window. Once the channel becomes idle, the node waits for another DIFS period before it starts to decrement its counter after each idle slot. When the backoff number reaches to zero, the node transmits its packet. When the receiver finishes its receiving, it waits for a shorter period SIFS and then sends back to the sender an ACK packet to inform the sender that the transmission is successful. If the sender hasn't received the ACK for a specified timeout or if it finds out some other node is transmitting a packet on the channel, the sender doubles its contention window CW and chooses a random number in the range $\left[0, \mathrm{CW}-1\right]$. **Figure 1** shows that the IEEE 802.11 adds two more signaling packets: the request to send (RTS) and the clear to send (CTS). When sending (RTS) to the destination node, the length of the transmission is attached; hence every node receiving this packet stores this information in a local variable named network allocation vector (NAV). After waiting a SIFS, the destination node replies with a CTS packet. This CTS packet also contains the duration of the transmission, therefore any node hearing this packet will set its NAV. All nodes within the range of the source node and the destination node are informed that the medium is allocated. The sender node, after waiting for SIFS, starts the data transmission. Then, the receiver node, after another SIFS, sends back the acknowledgement (ACK) packet. Afterwards, when the transmission is over, the NAV in each node marks the medium as free, and the process can start.

## 4. Our Improvement

The replay attack is an easy DoS attack which can be produced by a malicious MN through two basic operations. The first operation is the record of listened valid messages. The second is the resend of the recorded valid messages. Indeed, for a given communication between two MNs in the network, the replay attacker intercepts messages sent to destination MN and re-sends them later within a valid timestamp discrepancy $\Delta t$, independently, to any encryption mechanisms used by the sender MN. So the standard timestamp concept is not enough to limit impact of this type of DoS attacks on network performance.

The **Figure 2** illustrates a typical replay attack scenario where malicious MN, in the first step, intercepts and records signed messages listened from sender MN *S*. In second step and after a waiting time, within the timestamp discrepancy interval $\Delta t$, the attacker MN re-sends the stored signed messages, towards the receive MN *D*. As a result, all re-send messages by the replay attacker that verify the timestamp discrepancy present an overhead of messages which impact directly the network performance.

Recent works [22-24] are still using, in the process of message signature, a *prefixed timestamp discrepancy* $\Delta t$ negotiated in the step of encryption key exchange [25]. This choice of static timestamp gives a greatest weakness due to its independence on MN characteristics and duration of communication. Indeed, as shown in **Figure 3**, the replay attacker intercepts and stores the valid signed message before the end of time interval $\Delta t - \tau$. Thereafter, he achieves its attack by re-sending the previous stored messages in the dead time denoted $\tau$.

In this section, we present an enhanced timestamp discrepancy aiming to limit the impact of duplicated valid messages injected by a replay attacker intercalated between a pair of communicated MNs. Our approach has the advantage not to require any additional functions because it only based on the existing parameters defined in the MAC layer of the IEEE 802.11 standard. Our timestamp approach estimates approximately the date when the signed message is received and processed by a destination MN. Moreover, this estimation is a lightweight calculation and it is based on the standard parameters of 802.11 MAC layer. Referring to the **Figure 1**, the sender MN begins communication after receiving the CTS message sent by the receiver MN. In the same time, the neighbors MNs update their NAV parameter to defer

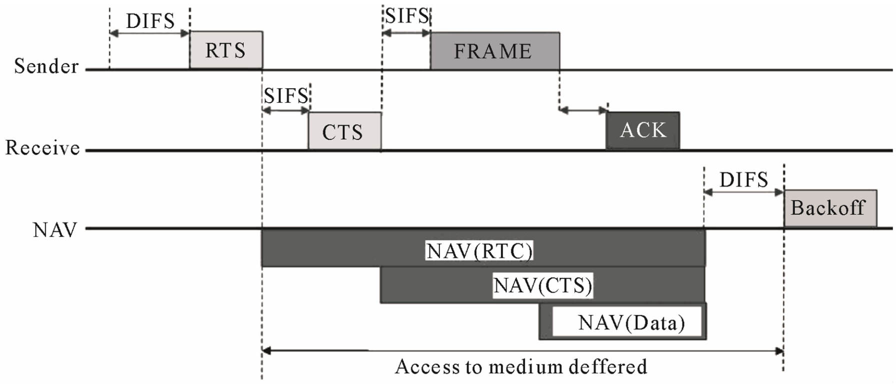

**Figure 1. RTS/CTS mechanism in IEEE 802.11.**

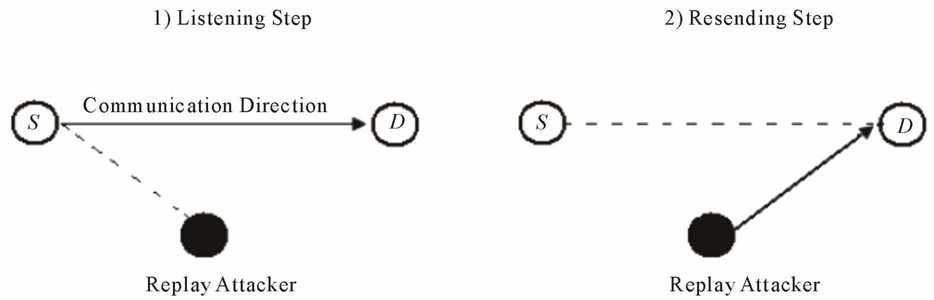

**Figure 2. Typical scenario of replay attack.**

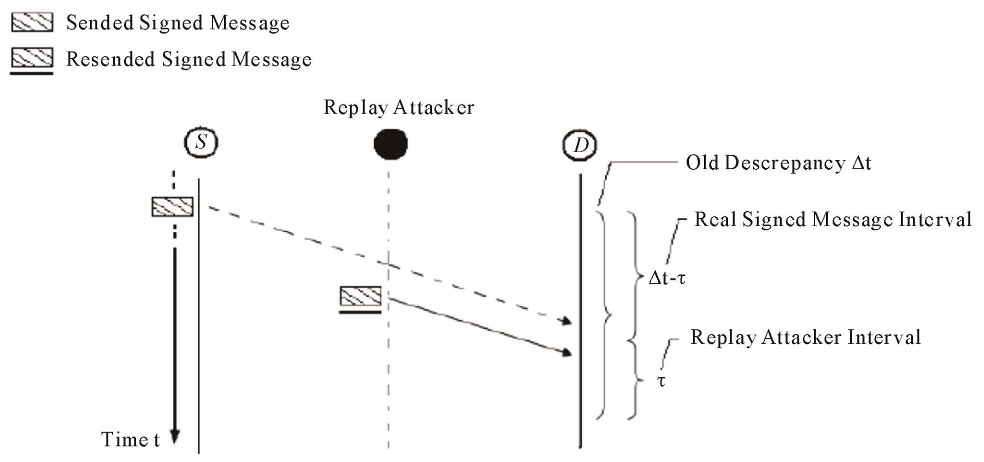

**Figure 3. Vulnerability with the classical timestamp discrepancy.**

access (DA) to the communication medium to avoid collisions. So, a sent signed message from a sender MN should arrive, to the receiver MN, and be processed before the NAV time expiration. The NAV expiration is delimited by the two messages: RTS (sent by the sender MN) and CTS (sent by the receiver MN). This means that the maximum time for a signed message to reach destination is the total time including NAV time plus processing times at the sender and receiver MNs.

Based on this observation, we can define the enhanced timestamp discrepancy between two given communicating MNs, *S* and *D* (See **Figure 4**) as follow [26]:

$$\Delta t_{\text{dynamic}}\left(S,D\right)=T_S+NAV\left(CTS\right)+T_D \qquad (1)$$

where:

- $T_S$ is the time to process message at MN *S*.
- $T_D$ is the time to process message at MN *D*.
- $NAV\left(CTS\right)$ is the time duration of communication between sender (*S*) and receiver (*D*) MNs.

In the following part of this work, in order to show the importance of our proposed improvement, we suppose that the communicating MN clocks are synchronized. This is a necessary condition for a replay attacker to re-send valid signed messages [22]. The times $T_S$ and $T_D$ represent respectively the total time at two MNs *S* and *D* including times of buffering and CPU processing. In the literature, buffering and CPU processes are respectively represented by the queuing and service systems. Precisely, the model that represents these two systems is an M/M/1 model [27], characterized by the following assumptions:

1) The messages arrive according to a Poisson process with a total *average arrival rate* $\lambda$ (*i.e.* arrival messages/sec).

2) The receiver MN (that plays the role of a single server characterized by an exponential service times, by an unlimited FIFO (or not specified queue) and by an unlimited messages population. We denote *the average service rate* at the receiver MN by $\mu$.

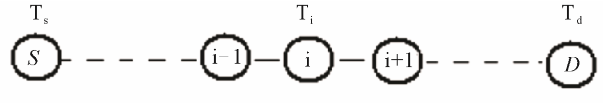

**Figure 4. Typical path between sender and receiver MNs.**

By supposing that MNs in MANET having the same characteristics, we can consider that $T_i=T_j=T$. So, the total time including queering and service times according to the M/M/1 model, at each MN, is given by the following formula:

$$T=\frac{1}{\mu-\lambda} \qquad (2)$$

Consequently, the Equation (1) becomes as follow:

$$\Delta t_{\text{dynamic}}\left(S,D\right)=\frac{2}{\mu-\lambda}+NAV\left(CTS\right) \qquad (3)$$

Based on the Equation (3), we can define a *local discrepancy timestamp* between two closed MNs (or neighbor MNs) in MANET as the average of *total discrepancy timestamp* $\Delta t_{\text{dynamic}}\left(S,D\right)$ divided by the number of hop count, that we denoted *N*, between S and D nodes. So, the local discrepancy timestamp $\delta t_{\text{local}}\left(i,i+1\right)$ between nodes $i$ and $i+1$ (see **Figure 4**) is defined as follow:

$$\delta t_{\text{local}}\left(i,i+1\right)=\frac{\Delta t_{\text{dynamic}}\left(S,D\right)}{N} \qquad (4)$$

In the next section, we proceed to apply our proposed approach on a two given communicating MNs in MANET, using 802.11 MAC layer to allow medium to exchange their messages. Our approach is integrated in the standard 802.11 MAC Layer without any additional parameters or extra processing costs at MNs in the network.

## 5. Simulation and Result

### 5.1. Simulation Environment

To improve the impact of our proposed timestamp concept, we simulated a local replay attack when a replay attacker is intercalated between two closed MNs in a MANET (each MN is in the transmission range of the other MN). Moreover, to have the same conditions of simulation, we assumed that all MNs in the network have the same characteristics of buffering ($\lambda$) and processing ($\mu$) with a stable M/M/1 system, *i.e.* the rate service is greater than the arrival rate. That's why we choose the values 30 and 33 for $\lambda$ and $\mu$ respectively for all MNs in the MANET.

In the next sub-section we proceeded to a comparison between two scenarios of communication with the *same replay attack behavior*. The first scenario is called a *classic scenario* where the communicated MNs use the classical timestamp discrepancy. The second scenario is called an *enhanced scenario* that uses our enhanced timestamp discrepancy. This comparison study is carried out in the Network Simulator (NS2) platform [28]. The communicating MNs, in a network, uses an UDP traffic to exchange data during a total time of simulation equal to 150 seconds. Moreover, we suppose that the MNs are homogenous in terms of transmission range (*i.e.* all MNs have a same transmission range equal to 250 m), and in order to show the effect of our approach, we have neglected the mobility produced by the free move of MNs. Finally, the considered replay attack interval when the attacker performs the attack is defined, in seconds, by the interval (100, 150).

### 5.2. Result and Discussion

To achieve a replay attack, the MN of the attacker requires a high performance, in terms of buffering $\lambda_r$ and processing $\mu_r$, compared to the ordinary MNs in the network. For this end, we have taken in our simulation the malicious behavior of the replay attacker when it's varying their proper parameters $\lambda_r$ and $\mu_r$. Precisely, to study the impact of each parameter on our enhanced timestamp discrepancy, we fixed, in first time, the parameter $\mu_r$ and we varied $\lambda_r$. In second time, we fixed the parameter $\lambda_r$ and we varied $\mu_r$.

By fixing $\mu_r$ at 33 and varying $\lambda_r$, **Figure 5** provides a light enhancement of our proposed timestamp (*red line*) discrepancy comparing to the old timestamp discrepancy (*black line*). Indeed, for all values of $\lambda_r$ (5, 10, 15, 20, 25, 30, 40, 45 and 49), the enhanced scenario that implements the dynamic timestamp discrepancy have the same behavior compared to the classic scenario with a limit of 2% approximately of the messages retransmitted and injected by the replay attacker. According to this result, it can be seen that our solution limits the number of the injected messages by the replay attacker even it changes the $\lambda_r$ parameter.

According to **Figure 6**, it can be seen that our enhanced scenario gives good result when the replay attacker changes its processing parameter. Indeed, for all values of $\mu_r$ (50, 55, 60, 65, 70, 75, 80, 85, 90, 95 and 100), the enhanced scenario (*red line*) that implement the dynamic timestamp discrepancy keeps the same behavior as the classic scenario (*black line*) with more rigorous limitation of injected messages. In particular, our approach

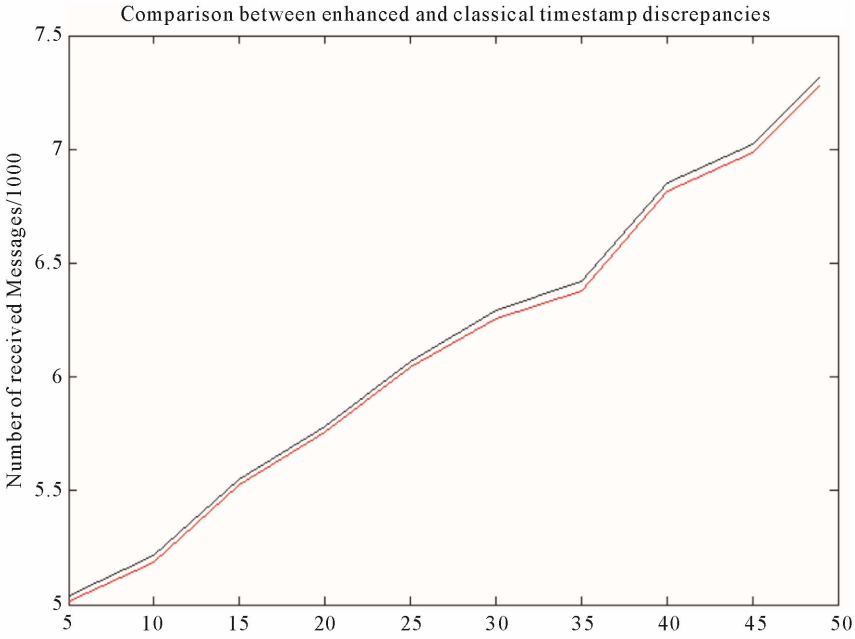

**Figure 5. Enhanced timestamp discrepancy when replay attacker varies $\lambda_r$.**

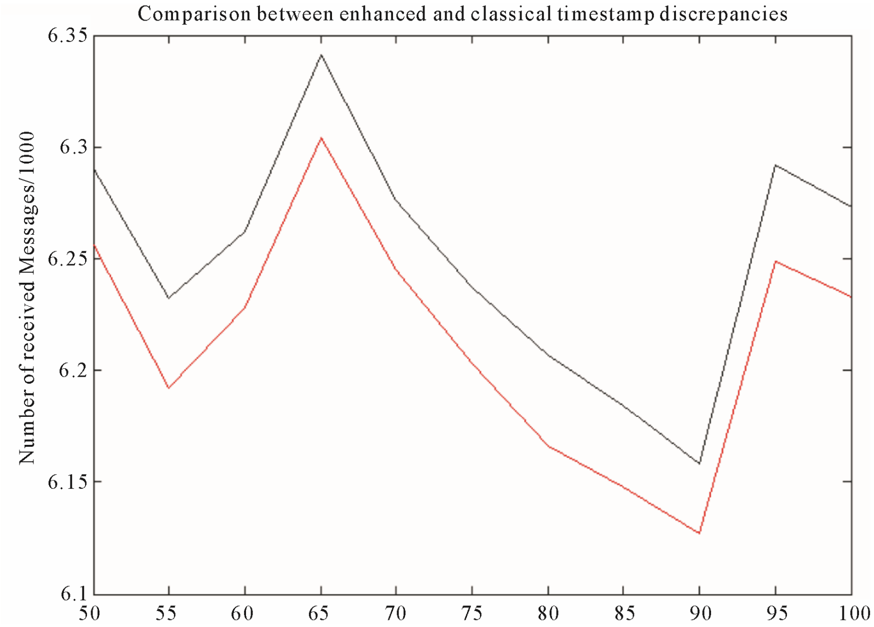

**Figure 6. Enhanced timestamp discrepancy when replay attacker varies $\mu_r$.**

gives a better reduction of replay attacker messages at points where $\mu_r$ takes the following values: 55, 80 and 95.

According to this result, we can say that our solution is reactive and watchful when the replay attacker changes its processing parameter $\mu_r$ .

Based on the above results, we conclude that our proposed timestamp discrepancy presents two big advantages: The first, it reduces the unwanted messages injected by a replay attacker. The second it takes into consideration the replay attacker behavior even when changing its proper parameters of buffering and processing.

## 6. Conclusions and Perspectives

In this paper, we propose a dynamic timestamp discrepancy to limit the impact of replay attacks in MANETs. Our approach reduces the number of unwanted messages injected by the replay attacker comparing to the classic timestamp discrepancy based on a fixed threshold defined at the beginning of a communication.

Therefore, the overhead produced by those types of attacks is reduced which increase the MANET performance. As a future work, we plan to study the MANET performance in the case where we combine our timestamp approach, at 802.11 MAC Layer, with the existing routing protocols. The routing protocols that we consider are AODV (*with reactive nature*) [29] and OLSR [30] (*with proactive nature*).

## REFERENCES

[1] J. Macker, “Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations,” Internet Engineering Task Force (IETF), Network Working Group (RFC 2501), January 1999.

[2] B. Wu, J. Chen, J. Wu and M. Cardei, “A Survey of Attacks and Countermeasures in Mobile Ad Hoc Networks, in Wireless Network Security,” In: Y. Xiao, X. Shen and D.-Z. Du, Eds., *Signals and Communication Technology*, Springer, 2007.

[3] J. Bellardo, S. Savage and D. Medina, “802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions,” *Proceedings of the USENIX Security Symposium*, Washington DC, August 2003, pp. 15-27.

[4] I. Aad, J. Hubaux and E. W. Knightly, “Impact of Denial of Service Attacks on Ad Hoc Networks,” *IEEE/ACM Transactions on Networking*, Vol. 16, No. 4, 2008, pp. 791-802. doi:10.1109/TNET.2007.904002

[5] K. Pelechrinis, M. Iliofotou and S. V. Krishnamurthy, “Denial of Service Attacks in Wireless Networks: The Case of Jammers,” *IEEE Communication Surveys and Tutorials*, Vol. 13, No. 2, 2011, pp. 245-257. doi:10.1109/SURV.2011.041110.00022

[6] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Standards 802.11, 1997.

[7] P. Syverson, “A Taxonomy of Replay Attacks,” *Proceedings of the Computer Security Foundations Workshop* (*CSFW*97), 1994, pp. 187-191.

[8] S. Malladi, J. A. Foss and R. B. Heckendorn, “On Preventing Replay Attacks on Security Protocols,” *International Conference on Security and Management*, June 2002, pp. 77-83.

[9] J. V. E. Molsa, “Increasing the DoS Attack Resiliency in Military Ad Hoc Networks,” *Proceedings of IEEE MILCOM*, Atlantic City, 2005, pp. 1-7.

[10] Q. Gu, P. Liu, S. Zhu and C.-H. Chu, “Defending Against

Packet Injection in Unreliable Ad Hoc Networks," Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM 05), 28 November-2 December 2005.

[11] Y. Xie and S. Yu, "Monitoring the Application-Layer DDoS Attacks for Popular Websites," *IEEE/ACM Transactions on Networking*, Vol. 17, No. 1, 2009, pp. 15-25. doi:10.1109/TNET.2008.925628

[12] R. M. Needham and M. D. Schroeder, "Using Encryption for Authentication in Large Networks of Computers," *Communications of the ACM*, Vol. 21, No. 12, 1978, pp. 993-999. doi:10.1145/359657.359659

[13] D. E. Denning and G. M. Sacco, "Timestamps in Key Distribution Protocols," *Communications of the ACM*, Vol. 24, No. 8, 1981, pp. 533-536. doi:10.1145/358722.358740

[14] T. H. Clausen, C. Adjih, P. Jacquet, A. Laouiti, P. Muhltahler and D. Raffo, "Securing the OLRS Protocol," *Proceedings of IFIP Med-Hoc-Net*, June 2003.

[15] A. Hafslund, A. Tnnesen, R. B. Rotvik, J. Andersson and O. Kure, "Secure Extension to the OLSR Protocol," *Proceedings of the OLSR Interop and Workshop*, San Diego, 2004.

[16] J. Suet and H. N. Liu, "Protecting Flow Design for DoS Attack and Defense at the MAC Layer in Mobile Ad Hoc Network," International Conference, ICAIC, Xi'an, Vol. 224, Part 1, 2011.

[17] S. Xu and T. Saadawi, "Revealing the Problems with 802.11 Medium Access Control Protocol in Multi-Hop Wireless Ad Hoc Networks," *Computer Networks*, Vol. 38, No. 4, 2002, pp. 531-548. doi:10.1016/S1389-1286(01)00273-0

[18] V. Gupta, S. Krishnamurthy and M. Faloutsos, "Denial of Service Attacks at the MAC Layer in Wireless Ad Hoc Networks," *MILCOM Proceedings*, Anaheim, Vol. 2, pp. 1118-1123.

[19] T. Farooq, D. L. Jones and M. Merabti, "MAC Layer DoS Attacks in IEEE 802.11 Networks," *The* 11*th Annual Conference on the Convergence of Telecommunications*, *Networking and Broadcasting* (*PGNet* 2010), Liverpool, 2010. http://www.cms.livjm.ac.uk/pgnet2010/MakeCD/Papers/2010063.pdf

[20] F. Xing and W. Wang, "Understanding Dynamic Denial of Service Attacks in Mobile Ad Hoc Networks," *Proceedings of the* 2006 *IEEE Conference on Military Communications* (*MILCOM*'06), Washington DC, 25-28 September 2006.

[21] J. Sobrinho, R. Haan and J. Brazio, "Why RTS-CTS Is Not your Ideal Wireless LAN Multiple Access Protocol," *Proceedings of IEEE Wireless Communications and Networking Conference*, New Orleans, 2005.

[22] D. Raffo, "Security Schemes for the OLSR Protocol for Ad Hoc Networks," Ph.D. Thesis, University Paris 6—INRIA Rocquencourt, 2005.

[23] E. Winjum, A. M. Hegland, Ø. Kure and P. Spilling, "Replay Attacks in Mobile Wireless Ad Hoc Networks: Protecting the OLSR Protocol," *Proceedings of International Conference on Networking* (*ICN* 2005), Springer-Verlag, Volume 3421/2005, 2005, pp. 741-479.

[24] B. Vaidya, M. Denko and J. R. Rodrigues, "Security Mechanism for Voice over Multipath Mobile Ad Hoc Networks," *Journal of Wireless Communications and Mobile Computing*, Vol. 11, No. 2, 2011, pp. 196-210. doi:10.1002/wcm.948

[25] D. E. Denning and G. M. Sacco, "Timestamps in Key Distribution Protocols," *Magazine Communications of the ACM*, Vol. 24, No. 8, 1981.

[26] N. Enneya, A. Baayer and M. El koutbi, "A Dynamic Timestamp Discrepancy against Replay Attacks in MANET," *Communications in Computer and Information Science* (*CCIS* 254), Springer-Verlag, 2011, pp. 479-489.

[27] D. Gross, J. F. Shortle, J. M. Thompson and C. M. Harris, "Fundamentals of Queueing Theory Book," 4th Edition, Wiley Series in Probability and Statistics, 2008.

[28] The Network Simulator (NS-2), 2012. http://www. is*i*.*e*du/nsnam/

[29] T. Clausen and P. Jacquet, "Ad Hoc On-Demand Distance Vector (AODV) Routing," RFC 3561, July 2003. http://www.ietf.org/rfc/rfc3561.txt

[30] T. Clausen and P. Jacquet, "RFC 3626: The Optimized Link-State Routing Protocol," Internet Engineering Task Force (IETF) Request for Comments, 2003.