Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 7, July 2013, pg.427 – 433

RESEARCH ARTICLE

Implementation of SAODV and TAODV Adhoc Secure Routing Protocols

T Nagarjuna[1], C.C. Kalyan Srinivas[2], K R Arjun Reddy[3]

[1]Assistant Professor, Department of Computer Science and Engineering, KMM Institute of Technology and Science, Tirupati, India

[2]Assistant Professor, Department of Computer Science and Engineering, KMM Institute of Technology and Science, Tirupati, India

[3]M.Tech Student, Department of Computer Science and Engineering, KMM Institute of Technology and Science, Tirupati, India

Email: arjun.reddykmmits@gmail.com

Abstract— Mobile Ad-hoc Networks (MANETs) allow wireless nodes to form a network without requiring a fixed infrastructure. Early routing protocols for MANETs failed to take security issues into account. Subsequent proposals used strong cryptographic methods to secure the routing information. In the process, however, these protocols created new avenues for denial of service (DoS). Consequently, the trade-off between security strength and DoS vulnerability has emerged as an area requiring further investigation. It is believed that different trust methods can be used to develop protocols at various levels in this trade-off. To gain a handle on this exchange, real world testing that evaluates the cost of existing proposals is necessary. Without this, future protocol design is mere speculation. In this paper, we give the first comparison of SAODV and TAODV, two MANET routing protocols, which address routing security through cryptographic and trust-based means respectively. We provide performance comparisons on actual resource-limited hardware. Finally, we discuss design decisions for future routing protocols.

Key Terms: - mobile; ad-hoc; security; routing; cryptography; trust-based; performance

I. Introduction

In traditional wireless networks, a base station or access point facilitates all communications between nodes on the network and communications with destinations outside the network. In contrast, MANETs allow for the formation of a network without requiring a fixed infrastructure. These networks only require that nodes have interoperable radio hardware and are using the same routing protocol to route traffic over the network. The lessened requirements for such networks, along with the ability to implement them using small, resource-limited devices has made them increasingly popular in all types of application areas. For example, MANET-based sensor networks have been proposed to assist in collecting data on the battlefield.

Since there is no fixed infrastructure, the nodes in the network forward traffic for one another in order to allow communication between nodes that are not within physical radio range. Nodes must also be able to change how they forward data over the network as individual nodes move around and acquire and lose neighbors, i.e., nodes within radio range. Routing protocols are used to determine how to forward the data as well as how to adapt to topology changes resulting from mobility. Initial MANET routing protocols, such as AODV [18], were not designed to withstand malicious nodes within the network or outside attackers nearby with malicious intent. Subsequent protocols and protocol extensions have been proposed to address the issue of security. Many of these protocols seek to apply cryptographic methods to the existing protocols in order to secure the information in the routing packets. It was quickly discovered, however, that while such an approach does indeed prevent tampering with the routing information, it also makes for a very simple denial of service (DoS) attack [10]. This attack is very effective in MANETs as the devices often have limited battery power in addition to the limited computational power. Consequently, this type of DoS attack allows for an attacker to effectively shut down nodes or otherwise disrupts the network.

The trade-off between strong cryptographic security and DoS has become increasingly important as MANET applications are developed which require a protocol with reasonable security and reasonable resistance to DoS, a kind of middle-ground. It has been suggested that various trust mechanisms could be used to develop new protocols with unique security assurances at different levels in this trade-off [5, 27]. However, the arguments for this have been purely theoretical or simulation-based. Determining the actual span of this trade-off in real world implementations is of\ utmost importance in directing future research and protocol design.

It is in this context that this paper considers two proposed protocol extensions to secure MANET routing. The first, SAODV [25], uses cryptographic methods to secure the routing information in the AODV protocol. The second, TAODV [15], uses trust metrics to allow for better routing decisions and penalize uncooperative nodes. While some applications may be able to accept SAODV's vulnerability to DoS or TAODV's weak preventative security, most will require an intermediate protocol tailored to the specific point on the DoS/security trade-off that fits the application. The tailored protocols for these applications will also require performance that falls between that of SAODV and TAODV. Understanding how the SAODV and TAODV protocols (which are on the boundaries of the DoS/security trade-off) perform on real hardware, and to what extent there exists a performance gap is a prerequisite for being able to develop the intermediate protocols. Such evaluations not only required for developing intermediate protocols, but also for determining the direction for development of new trust metrics for ad-hoc networks. In this paper we provide the first performance evaluations for these protocols on real world hardware.

II. RELATED WORK

Several different protocols have been proposed for ad-hoc routing. The earlier protocols such as DSDV [19], DSR [11], and AODV [18] focused on problems that mobility presented to the accurate determination of routing information. DSDV is a proactive protocol requiring periodic updates of all the routing information. In contrast, DSR and AODV are reactive protocols, only used when new destinations are sought, a route breaks, or a route is no longer in use.

As more applications were developed to take advantage of the unique properties of ad-hoc networks, it soon became obvious that security of routing information was an issue not addressed in the existing protocols. In [13], Lundberg presents several potential problems including node compromise, computational overload attacks, energy consumption attacks, and black hole attacks. Deng et al. Further discuss energy consumption and black hole attacks along with impersonation and routing information disclosure in [3]. Jacobson et al. categorize attacks as manipulation of routing information and exhaustive power consumption, and provide detailed treatments of many characteristic attacks in [10].

While research has focused on "lightweight" security mechanisms, some proposed protocols use more expensive asymmetric cryptography. In [26], Zhou and Haas present a multi-path protocol extension that uses threshold cryptography implement the key management system. It requires some nodes to function as servers and an authority to initialize these servers. Zapata and Asokan propose SAODV [25], a secure version of AODV, which uses digital signatures and hash chains to secure the routing messages.

In [22], Pissinou et al. propose a trust-based version of AODV using static trust levels. The same authors then extend this protocol in [7] to thwart multiple colluding nodes. Neither of these addresses securing the trust exchanges, or the overhead involved. Li et al. introduce a trust-based variant of AODV in [12] that secures the trust information. However, their protocol requires an intrusion detection system in the network. Finally, Meka et al. propose a third trusted AODV with a simple method of evaluating trust even without source routing [15].

About how attackers tend to attacks:

Unfortunately, these passwords are broken mercilessly by intruders by several simple means such as masquerading, Eaves dropping and other rude means say dictionary attacks, shoulder surfing attacks, social engineering attacks [7][1].To mitigate the problems with traditional methods, advanced methods have been proposed using graphical as passwords. The idea of graphical passwords first described by Greg Blonder (1996). For Blonder, graphical passwords have a predetermined image that the sequence and the tap regions selected are interpreted as the graphical password. Since then, many other graphical password schemes have been proposed. This systematic examination provides a comprehensive and integrated evaluation of PCCP covering both usability and security issues, to advance understanding as is prudent before practical deployment of new security mechanisms.

Our work in this paper considers the asymmetric cryptography and trust-based extensions to AODV presented in [25] and [15] respectively and show a real world comparison of the performance of the two protocols. Our results suggest that new protocols can be developed which take advantage of the best features of both types of protocols, and which share aspects of each security model.

III. PROTOCOL OVERVIEWS

Due to space considerations, the reader is referred to for descriptions of the AODV, SAODV, and TAODV protocols respectively.

IV. EXPERIMENTAL SETUP

Since ad-hoc networking's most promising applications make use of small, resource constrained devices that are significantly different from today's ever faster desktop computers, special attention must be paid to the trade-off between strong cryptographic security and DoS. While theoretical analysis or simulation may give helpful hints on the relative efficiency of different approaches, only real world implementation and performance testing can give a concrete picture of the actual width of this spectrum. Such measurements provide the necessary information to determine which protocols are suitable for specific applications. In addition, the results can then be used to guide the design of novel protocols better suited to particular deployment situations.

In order to get an understanding for the real world performance of the AODV, SAODV, and TAODV protocols, we have implemented each of them on real hardware and measured their performance. In this section we detail the setup for the experiments used to acquire these measurements. We first describe the supporting hardware and software setup for our implementations. We then present details on the actual implementation for each of the three protocols. Finally we detail the design of the experiments used to evaluate the protocols and explain why these tests are more relevant than other more common metrics.

4.1 Hardware and Software Setup

Since many of the application areas for ad-hoc networks include resource-limited devices, for example sensor networks, it is necessary to test these protocols on such devices. For our testing we used the Sharp Zaurus SL-5500 model palmtops. The SL-5500 contains a 206MHz Intel StrongARM processor, 64MB of DRAM, a 16MB Flash ROM, a 950 mAH lithium ion battery and both Compact Flash and Secure Digital card slots. In short, the Zaurus is as powerful as a desktop computer was a decade ago. With the rapid advances in technology, a device with these capabilities could become the embedded sensor network device of the near future. Regardless, they allow for an analysis using processors that are an order of magnitude out of step with today's conventional processors.

Each Zaurus was equipped with a LinksysWCF11 compact flash card for wireless communication. The Zauruses ran OpenZaurus [17] v3.5.4, an embedded version of Linux. In order to compile programs for the Zaurus we used a cross-compiler tool chain based on GCC v3.3.4. In addition, as described in Section 4.2, our code requires the OpenSSL [16] libraries. For this purpose, OpenSSL v0.9.7j was cross compiled and statically linked into executables where necessary. All cross-compiling was performed on a desktop running Slack ware Linux 11.0 [23].

4.2 Implementation

Our AODV implementation is the result of previous projects in this area. The implementation is designed to run on the Linux operating system. As with many other AODV implementations for Linux, it separates functionality into a kernel module and a user space daemon. The kernel module uses hooks in the net filter interface to send packet headers from the wireless interface to the user space daemon. The daemon then determines how to handle the packet. If the packet is a routing control packet, then the daemon processes the packet in accordance with the AODV specification. If instead the packet is a data packet, the daemon determines whether or not a route exists to the necessary destination. If there is a suitable route, the packet is flagged and the kernel module queues it to be sent out. If no route exists, the daemon begins route discovery. Once a route is found, the daemon enters the route into the kernels routing table. It then flags the packet (and any additional packets arriving during discovery) to be queued for transmission. The implementation is written completely in C.

In order to implement SAODV, it was necessary to have a library of cryptographic operations. We used OpenSSL for this purpose, and we developed a security library which wrapped much of OpenSSL's functionality into components appropriate for ad-hoc routing purposes. One particularly useful feature of the security library is that it allows easy use of several different OpenSSL contexts at once. For SAODV, this was useful as nodes must switch between signing, verifying, and hash chain operations rapidly to both send and receive routing messages. New data structures were added for SAODV's single signature extension and the necessary code was added to the message processing functions for **RREQ, RREP**, **HELLO**, and **RERR** messages. The design of the AODV implementation allowed SAODV functionality to be implemented while maintaining one binary with the ability to run both protocols.

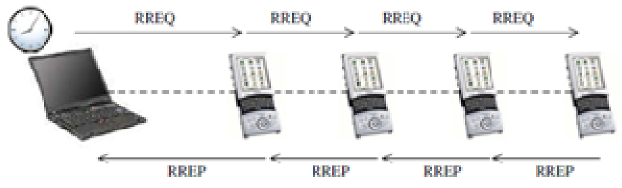

Fig.1. Network setup for round trip timing tests

Library is that it allows easy use of several different OpenSSL contexts at once. For SAODV, this was useful as nodes must switch between signing, verifying, and hash chain operations rapidly to both send and receive routing messages. New data structures were added for SAODV's single signature extension and the necessary code was added to the message processing functions for RREQ, RREP, HELLO, and RERR messages. The design of the AODV implementation allowed SAODV functionality to be implemented while maintaining one binary with the ability to run both protocols.

Implementing TAODV required many additions similar to those involved in SAODV. New data structures were used for the NTT as well as the extended messages and the new **R_ACK** message. Similarly, message handling functions were updated to use the extensions and take the appropriate actions. One challenge in implementing TAODV was counting packets sent, forwarded, or received for a particular route. While it intuitively seems to be something that should be implemented in the kernel module that is already tied into the netfilter framework, this would require extra data exchange between the kernel module and the daemon. Since our implementation already passes packet headers to the daemon for route discovery initiation and flagging, it was simply necessary to place the counting mechanism in the daemon.

Keeping track of the additional routing information required significant extension of our AODV implementation. The original implementation does not support any multi-path entries in the routing table. Modifying it to support such a setup for TAODV would have required rewriting significant amounts of the base AODV code. Instead, we implemented a multi-path capable routing table for use exclusively by the TAODV protocol. When a node initially discovers a route, or changes the active route to a particular destination, it merely copies the necessary entry to the daemon's local routing table and marks it as having been altered so that it is updated in the kernel's routing table at the next sync. This simplified the implementation using only a negligible amount of extra memory.

4.3 Testing Setup

There were two performance factors we were interested in for the purposes of this comparison. The first is the per-packet processing overhead. It is important to note that only CPU time was measured. Therefore this overhead reflects use of the processor by each protocol. In these tests we use AODV as a baseline. Thus, for SAODV we measure the time it takes to generate an SSE for **RREQ, RREP**, and **HELLO** messages. We also measure the time it takes for a node to verify an SSE for those same messages. For TAODV we measure how long it takes a node to generate or process and update **RREP** and **R_ACK** messages. Due to the fact that some of the operations we measure have a runtime less than the resolution of our timer (10ms as per the Linux kernel), we perform a large number of operations back-to-back per measurement. We then make multiple measurements.

Operation	Proc. Time (ms)	Std. Dev.
SSE generation	30.8	0.028
SSE validation	3.81	0.006

Table 1: SAODV Per-Packet Overhead Times

Our second performance metric is round trip time for route discovery. The justification for this metric lies in the fact that we are looking at securing the routing control packets. Once a route is established, data is forwarded with the same efficiency regardless of the routing protocol. Therefore, it is important to see how the per-packet overhead along with the increased packet size affect the time for route discovery. For this test, we measure the performance of AODV in addition to that of SAODV and TAODV. This is necessary because both AODV and TAODV will generate **RREP**s after fewer hops when the destination's neighbor responds, while SAODV requires that the destination itself responds. For our experiments, we used a five node network consisting of one laptop and four Zauruses as illustrated in Figure 1. We used the network sniffer ethereal [6] running on the laptop to measure the time elapsed from the sending of the **RREQ** to the receipt of the **RREP**. These individual measurements were also performed repeatedly as explained in Section 5.

V. RESULTS

5.1 Per-Packet Results

For the per-packet overhead tests, we measured the amount of processing time a node spends above and beyond that required for conventional AODV. All tests were performed on the Zauruses with only the necessary software running (i.e.,no graphical login manager, no X server, etc.). In the SAODV tests, we measure generation and validation of the SSE which requires hash computation and a digital signature/verification. The hash function used for these tests was MD5 and the digital signature/verification was performed using a 512-bit RSA key pair. There were 1000 operations run per measurement and 1000 measurements overall. Table 1 shows the results of our SAODV tests.

Consequently, in order to send a **RREQ, RREP**, or **HELLO** message, the node spends 30.8 milliseconds generating the SSE. The significant impact on performance occurs in generating the SSE for **HELLO** messages since they are sent periodically. According to the AODV specification, a node should send a **HELLO** message every **HELLO INTERVAL** milliseconds unless it has broadcast any messages during the previous interval. This means that only **RREQ** and **RERR** messages could prevent sending a **HELLO** message, as all other messages are unicast. Obviously, this can place a significant burden on each node.

Operation	Proc. Time (ms)	Std. Dev.
RREP/HELLO send	0.0453	0.002
RREP/HELLO processing	0.0452	0.002
R ACK send	0.193	0.004
R ACK processing	0.297	0.005

Table 2 : TAODV Per-Packet Overhead Times

Since SAODV requires that each message with a SSE is validated before any further processing takes place, each **RREQ** and **RREP** gets delayed 3.8 milliseconds at each hop which forwards it. In addition, **HELLO** messages take the same amount of time to be validated. While nodes are supposed to let **ALLOWED HELLO LOSS * HELLO INTERVAL** milliseconds pass before deciding a link is broken and a neighbor should be removed from its routing table, it is conceivable that on a node with several neighbors and a large amount of data to forward, route status may fluctuate for some neighbors whose **HELLO** packets get delayed in validation.

In TAODV, we measure the per-packet overhead for **RREP**, **HELLO**, and **R_ACK** messages. The system-wide parameters discussed in [15] do not influence the overhead of TAODV for any of the tests we performed. However, it was necessary to fix these values to allow for the calculation of RSV. For all TAODV tests we used the following system-wide parameter values: $i = 0.8$, $pl = 0.6$, $ph = 0.4$, $pc = 0.2$, $\alpha1 = 0.4$, $\alpha2 = 0.4$, and $\alpha3 = 0.2$. Due to the very small running time of the operations, one million operations were performed per measurement and 5000 measurements were taken. Table 2 shows the results for the TAODV tests.

As the results show, there is much less per-packet overhead for TAODV when compared to SAODV. The main source of overhead involved the **R_ACK** packets. Since the **R_ACK** packets are new packets rather than packet extensions, it was necessary to allocate a packet buffer in the message sending system of our implementation each time a **R_ACK** packet was to be sent. With other messages that were extended, the packet buffer was already allocated and the extension was simply written into free space at the end. This difference contributed significantly to the 0.193ms overhead for sending the **R_ACK** message.

The overhead for processing the **R ACK** message was almost completely due to the recalculation of the OTV and RSV values. The TAODV implementation used double primitives for all calculations in order to keep with the protocol description in [15]. However, this affects the performance since the SA-1110 processor in the Zaurus has only integer arithmetic units. For systems with less computational power than the Zaurus' these results suggest that it may be necessary to rewrite trust-based metrics into their equivalent using integer arithmetic instead.

5.2 Round Trip Results

The round trip tests for route discovery were performed for all three protocols. This was particularly important due to the differences in which node sends the **RREP** as described in Section 4.3. Due to the nature of the measurements, only one route discovery operation could be executed per measurement. Overall 5000 of these individual measurements were performed. Table 3 shows the results of the tests

Protocol	Round Trip Time	Std. Dev.
AODV	(ms)	0.765
SAODV	324.732	7.22
TAODV	152.780	0.863

Table 3: Round Trip Times

These results show that SAODV is indeed a significantly more expensive protocol. Specifically, SAODV takes 2.35 times as long as conventional AODV to get a **RREP** back to a **RREQ** originator. This is due, in part, to the added cryptography and increased message size. This is also due to the inability of intermediate nodes to respond to **RREQ**s. Traversing the additional hop in both directions adds to the latency. While we did not implement the DSE, this should not have a large effect on the average route discovery since a destination now has to generate two digital signatures for a **RREP**. In addition, DSE only addresses the overhead incurred by intermediate nodes not responding to **RREQ**s. There still is overhead from the added cryptography and increased message size which implementing DSE will not solve.

The results also show that the use of SAODV will require adjustments to the recommendations for configurable parameters in AODV. This is missing from the current draft standard for SAODV. For example, the current suggested **NODE TRAVERSAL TIME** is 40ms which results in **NET TRAVERSAL TIME** being set to 1400ms. The value of **NET TRAVERSAL TIME** serves as the timeout for RREQ messages. Consequently, as per the results above, if these parameters were not adjusted, nodes would have problems discovering routes of length greater than seventeen hops. In some applications this may not cause problems. However, in certain applications such as large area sensor networks, routes of this length or greater would not be unreasonable to expect.

TAODV, on the other hand, takes only 1.11 times as long as AODV. This shows that the trust-based calculations and additional information exchange can be used without incurring the overhead of SAODV. While there is some expense for the trust calculations, it is not nearly as expensive as the cryptographic operations. The results show that TAODV is indeed at the opposite end of the trade-off from SAODV. This is due to the fact that the TAODV information itself in each packet is not secured.

Overall, the results show that there is indeed a wide spectrum in the tradeoff between cryptographic security and DoS. By adding an appropriate lightweight security mechanism to secure the trust information in the routing packets, a hybrid protocol can be created which is less expensive than SAODV and more secure than TAODV. Future protocol designs should seek to use various new combinations of smarter, trust-based metrics and lightweight security mechanisms in order to develop hybrid protocols across this spectrum.

VI. Conclusion

In this paper, we have compared the SAODV and TAODV protocols for securing adhoc network routing. We presented the results of implementation and evaluation of both protocols on real resource-limited hardware. The expected difference between the two protocols was shown to be consistent with this real world scenario. These experiments showed that there is significant room between the two protocols for a secure hybrid protocol to be developed which takes advantage of the strongest points of both. Future work needs to delve further into the extensive body of work on various trust metrics. This includes the testing of other trust metrics for use in ad-hoc routing as well as developing the aforementioned hybrid protocols and testing their performance against the results presented in this paper. In addition, it is necessary to test the quality of the routing decisions produced by all of these protocols in a malicious environment.

References

[1] C. N.-R. Baruch Awerbuch, David Holmer and H. Rubens. An on-demand secure routing protocol resilient to byzantine failures. In ACM Workshop on Wireless Security (WiSe), September 2002.

[2] S. Buchegger and J.-Y. L. Boudec. Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks. In Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed and Network-based Processing. IEEE Computer Society, January 2002.

[3] H. Deng. Routing security in wireless ad hoc networks, 2002.

[4] P. Dewan and P. Dasgupta. Trusting routers and relays in ad hoc networks. In ICPPW ’03: Proceedings

of the 2003 International Conference on Parallel Processing Workshops, pages 351–358, 2003.

[5] L. Eschenauer, V. Gligor, and J. Baras. On trust establishment in mobile ad-hoc networks. Technical Report MS 2002-10, Institute for Systems Research, University of Maryland, MD, USA, October 2002.

[6] Ethereal - A Network Protocol Analyzer. http://www.ethereal.com/.

[7] T. Ghosh, N. Pissinou, and K. Makki. Collaborative trust-based secure routing against colluding malicious nodes in multi-hop ad hoc networks. In LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN'04). IEEE Computer Society, 2004.

[8] Y. Hu, D. Johnson, and A. Perrig. SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks. Ad Hoc Networks, I:175–192, 2003.

[9] Y. Hu, A. Perrig, and D. Johnson. Packet leashes: A defense against wormhole attacks in wireless adhoc networks. Technical report, Department of Computer Science, Rice University, December 2001.

[10] M. Jakobsson, S. Wetzel, and B. Yener. Stealth attacks on ad hoc wireless networks. In Proceedings of VTC, 2003, 2003.

[11] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, Mobile Computing, volume 353. Kluwer Academic Publishers, 1996.

[12] X. Li, M. Lyu, and J. Liu. A trust model based routing protocol for secure ad hoc networks. In Proceedings of the Aerospace Conference, 2004.

[13] J. Lundberg. Routing security in ad hoc networks, 2000.

[14] S. Marti, T. J. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In Mobile Computing and Networking, 2000.

[15] K. Meka, M. Virendra, and S. Upadhyaya. Trust based routing decisions in mobile ad-hoc networks. In Proceedings of the Workshop on Secure Knowledge Management (SKM 2006), 2006.

[16] OpenSSL. http://www.openssl.org/.

[17] OpenZaurus. http://www.openzaurus.org/.

[18] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing. July 2003.

[19] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications, pages 234–244, 1994.

[20] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar. SPINS: security protocols for sensor netowrks. In Mobile Computing and Networking, 2001.

[21] A. Pirzada and C. McDonald. Establishing trust in pure ad-hoc networks. In Twenty-Seventh Australasian Computer Science Conference (ACSC2004), 2004.

[22] N. Pissinou, T. Ghosh, and K. Makki. Collaborative trust-based secure routing in multihop ad hoc networks. In NETWORKING 2004, Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications, 2004.

[23] The Slackware Linux Project. http://www.slackware.org/.

[24] S. Yi, P. Naldurg, and R. Kravets. Security-aware ad hoc routing for wireless networks. In MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing, pages 299–302, New York, NY, USA, 2001. ACM Press.

[25] M. G. Zapata and N. Asokan. Securing ad hoc routing protocols. In WiSE '02: Proceedings of the ACM workshop on Wireless security. ACM Press, 2002.

[26] L. Zhou and Z. J. Haas. Securing ad hoc networks. IEEE Network, 13(6):24–30, 1999.

[27] C. Zouridaki, B. L. Mark, M. Hejmo, and R. K. Thomas. A quantitative trust establishment framework for reliable data packet delivery in manets. In SASN '05: Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks, pages 1–10, New York, NY, USA, 2005. ACM Press.