J Comput Virol (2008) 4:267–278
DOI 10.1007/s11416-007-0072-y

ORIGINAL PAPER

Improving security and performance of an Ad Hoc network through a multipath routing strategy

Hervé Aiache · François Haettel · Laure Lebrun · Cédric Tavernier

Received: 23 June 2007 / Revised: 31 October 2007 / Accepted: 21 November 2007 / Published online: 25 January 2008

Abstract Privacy and security solutions require today the protection of personal information so that it may not be disclosed to unauthorized participant for illegal purposes. It is a challenge to address these issues in networks with strong constraints such as Ad Hoc network. The security increase is often obtained with a quality of service (QoS) decrease. We propose in this paper a solution that provides at the anonymity, security to Ad Hoc network with a limited impact on QoS. This method could be efficient against some viral attacks. We also give some security proofs of our solution for Ad Hoc networks.

1 Introduction

Nowadays, security and privacy are becoming crucial in communication systems as the number attacks constantly increasing over the Internet. Our solution brings some security against viral attacks. In fact, personal information are requested by service providers from customers (e.g., digital stores, location services or bank access). This information, if not protected, is sensitive to even passive attackers. For data protection, integrity, authentication, confidentiality and non-repudiation are ensured using encryption, hash and MAC (Message Authentication Code). For communication protection, traffic source and destination, traffic paths and the type of traffic has to be protected. Since the 1980's, many efficient systems (e.g., [9,17,10]) ensured communication protection of network. Historically, they were proposed for wired networks and are mainly derived from the so-called Chaum's Mix concept [9].

In this context, with the growing interest in wireless networks and the increased integration of mobile and small computing user device in existing communications systems, the Research Community focused its effort on security and privacy to wireless Mobile Ad Hoc networks (MANETs). However, as mentioned, enabling privacy protection differs from providing security, specifically in the case of MANETs. In fact, securing data can be achieved through cryptographic techniques but control information such as node address, which are necessary to transmit information from source to destination, cannot be simply encrypted to achieve the network infrastructure basic purpose. Moreover, MANETs are more vulnerable than their wired homologue. First, because an attacker can easily eavesdrop remotely wireless transmission. Secondly, adding cryptographic protection to wireless Ad Hoc networks is difficult because they require high data rate exchange and because the cryptographic program must be run on low CPU and/or memory capabilities to compose the Ad Hoc infrastructure. In this context, many secured routing schemes have been proposed to protect the main Ad Hoc routing protocols, which are AODV [42], DSR [23], DSDV [41] or OLSR [8]. For example, SRP [43] was proposed to validate the integrity of the proposed route through DSR by assuming security associations between the source and the destination node. ARAN [46] tackled the same SRP issues by using public key cryptography and proposed mechanisms

The work of Hervé Aiache and Cédric Tavernier was supported by DISCREET, IST project no. 027679, funded in part by the European Commission's Information Society Technology 6th Framework Programme.

H. Aiache (✉) · F. Haettel · L. Lebrun · C. Tavernier
THALES Communications, Colombes, France
e-mail: Aiache.Herve@fr.thalesgroup.com

F. Haettel
e-mail: Haettel.Francois@fr.thalesgroup.com

L. Lebrun
e-mail: Lebrun.Laure@fr.thalesgroup.com

C. Tavernier
e-mail: Tavernier.Cedric@fr.thalesgroup.com

for non-repudiation of discovered route. Another interesting example is SAR [50], which introduced mechanisms to discover routes with a given security criteria. Other examples include AODV-S [49], Ariadne [21], SEAD [20,48] or SPREAD [34]. However, most of these solutions mainly focus on security issues and cannot be directly used for privacy protection through the definition of an anonymous routing scheme. For example, in some of these solutions, intermediary forwarding nodes are able to easily identify which nodes are communicating, simply by handling routing control messages: this does not meet node anonymity requirements. As preserving privacy for users becomes an important concern, the number of propositions oriented on anonymous routing increases for MANETs. Most of them are based on on-demand scheme, relying on reactive routing protocol approach. The anonymity schemes are mainly originated from Mix-net approach [9,10], which needs to be improved in term of performance to be suitable in the context of constrained network environments, such as MANETs. Strong attacks also come from malware intrusion. Such intrusion can directly compromise privacy. Our solution resists against such attacks.

Among the requirements, the protection of information plays a central role. The concept of anonymity is quite recent: users want that no one can determine with whom they communicate, implying the necessity to hide the user identity from eavesdroppers and to resist against traffic analysis. To protect information, data can be ciphered using a secret key cryptosystem which decreases the performance.

We propose an original solution inspired by network coding techniques and the McEliece public key cryptosystem (see [36]). This technique has the advantage to provide anonymity and confidentiality without affecting too much the quality of service. However we do not claim to guarantee perfect security (which is impossible) but rather an average security. We focus on a solution that ensures a relative security for information during a relatively limited period of time. The main objective of our solution is to discourage attackers by requiring too much effort to recover the information compared to the time when the information is relevant.

Our proposal assumes that a multipath routing or forwarding protocol is available. Multiple paths between a source and a destination ensure packet diversity, which improves anonymity: the information is scattered into several packets which belong to different flows. This strong requirement for network protocols allows us to detect intrusion and to protect information as described later in the document.

2 Related work

This section is structured as follows: first, an overview of current anonymous routing solutions is presented, attempting to improve crypto-functions to increase privacy protection. Secondly existing anonymous routing propositions are briefly described, focusing on network performances. Then properties are derived to adapt the presented solution to wireless Ad Hoc networks. Then, based on these properties, crucial issues are described to combine a high level of security, privacy and performance. Open issues are also discussed in the conclusion.

2.1 Approaches increasing privacy protection

ANODR [27–29] was one of the first on-demand anonymous routing protocol proposed in the literature for MANETs. ANODR protocol protects privacy by using reactive routing approach (i.e., on demand) to discover routes and a broadcast mechanism combined with receiver trapdoor assignment to ensure source identity anonymity. Moreover, ANODR relies on mechanisms similar to onion routing [5,6,11]: the source node creates a particular onion (called "boomerang onion") in the route request message (RREQ) flood packet to which each forwarding node in the route adds an encrypted layer to the RREQ and then the destination node uses this onion to send the route reply message (RREP) back to the source node. The source and destination nodes do not necessary know the forwarding node identifier. As mentioned, the destination node identifier (in the RREQ) is encrypted in a trapdoor, which can be only decrypted by the destination node. In this way, the source is able to establish an anonymous virtual circuit with the destination. Note that in comparison to SDDR [14], which was proposed to provide a weak location privacy and a route anonymity, the mechanisms proposed by ANODR for data transmission are more efficient. However, ANODR and SDDR do not really achieve node identity anonymity and strong location privacy. ASR [51], a variant of ANODR as demonstrated by Kong et al. [26], was proposed to improve and to reinforce privacy requirements. In fact, one of the main difference between ASR and ANODR concerns some change in the crypto-functions, such as one-time pad to replace AES or stronger cryptographic functions for encrypting routes. Nevertheless, ASR or ANODR by mainly focusing on privacy protection based on a reactive routing approach, faces some difficulty in terms of small mobile device capabilities (i.e., CPU and memory) and of Ad Hoc network resources. For example, when a node receives a RREQ, it should decrypt the trapdoor identifier with each key shared with other nodes. Another example is that each node composing the MANET has to regularly generate a public/secret key pair (for every RREQ). This kind of crypto-operation has a cost which is not negligible, and more specifically for small device limited in processing and memory capabilities. In this way, the main drawback of ASR and ANODR is efficiency, as pointed by [44]. Note also that these anonymous routing protocols are more adapted to

low-end device, as analyzed by Liu et al. [35]. As for ASR and ANODR, SDAR [1] is also based on a reactive routing scheme to implement the route discovery function. However, as attempted by ASR upon ANODR, SDAR improves anonymity protection through strong crypto-functions. SDAR relies on an onion routing scheme to provide route anonymity and data exchange privacy between the source and the destination node. In fact, SDAR, as for AnonDSR [45], uses a key management scheme that aims at establishing and collecting symmetric keys between the destination node and each intermediary forwarding nodes defining the discovered route and a global trapdoor that is managed between the source and the destination nodes. More specifically, in SDAR, a source node sends a RREQ message, to which, each intermediary node adds its encrypted identifier before forwarding it. The destination node is the only node that is able to decrypt the intermediary nodes identifiers contained in the RREQ. Then, the destination node uses these identifiers to create an onion-like object integrated in a RREP message and sends it back to the source node. Nevertheless, as for the improvement of ASR upon ANODR, it focuses on the reinforcement of crypto-functions to ensure anonymity and therefore presents some difficulty to provide efficiency. As analyzed by Liu et al. [35], SDAR induces an important communication overhead due to the size of messages. Note that the source and the destination node are able to identify the intermediary forwarding nodes of the discovered route. And, each intermediary forwarding node needs to perform a public key decryption, a public key encryption and to generate a signature for each RREQ. Moreover, in comparison to ANODR, which only requires a pair-wise key agreement between neighboring nodes to establish the anonymous circuit identifier,as mentioned before, SDAR needs to share, a symmetric session key with each intermediary forwarding nodes composing the discovered route. This partially explains that Liu et al. [35] estimate that SDAR, as for AnonDSR, is more suitable to high-end nodes that handle public key cryptography efficiently. Remark that SDAR, as AnonDSR, introduces a light difference to improve anonymous routing in MANET when compared to ASR or ANODR: it combines a reactive route discovery approach to a proactive MIX-net [3,5,30,40] scheme.

2.2 Solutions improving network performances

The above deviation from a pure on-demand routing approach is also tackled by MASK [52] to improve not only directly privacy protection but also anonymous routing performances in term of network capacity. MASK integrates a proactive neighbor detection protocol to create one-hop anonymous links with its neighbors prior to on-demand route discovery scheme to establish virtual anonymous circuit. This knowledge of the node neighborhood aims at reducing crypto-functions processing overheads naturally induced by a pure reactive approach (through its on-demand route discovery function). This proactive neighbor detection scheme is identity-free (as originally proposed globally by ANODR) and is performed through a pairing-based anonymous handshake [2] between any pair of neighbors. For key exchanges between a given node and its new detected neighbors, MASK relies on a three step handshake. Once the handshake procedure is ended, each pair of nodes shares a chain of secret keys and locally unique link identifiers pair that corresponds to the pseudonyms used during handshake. In the proactive neighbor detection part of MASK, HELLO messages are periodically sent to hold the pairing cryptographic materials. Then, MASK relies on classical pure on-demand route discovery scheme. As for ASR, intermediary forwarding node keeps an information state about previously RREQ message sent by a source node. Once it receives a RREP message, this information is used by each intermediary node to decide whether it should forward the information and to which node. However, MASK presents some drawbacks concerning privacy protection by focusing on the improvement of network capacity, contrary to ASR or SDAR. For example, as summarized by [44], the final destination appeared in plain text in each RREQ. Moreover, MASK relies on a tight synchronization between keys and pseudonyms of neighboring nodes. More recently, ODAR [47] has been proposed to tackle also performance issues at different levels, necessary to design a suitable anonymous routing solution for Ad Hoc networks: storage, processing and communication. This approach is based on a reactive routing approach and relies on Bloom filters, previously used in [7] to provide node, link, path anonymity and efficiency at once.

By definition, the wireless environment is constraining because communication performance largely depends on the quality of the radio transmission and on the capacity of the networked device. Errors bits or burst due to this transmission are common in such an environment, the capacity could be weak (50 kits/s) and the mobile devices like PDAs and laptops are not so efficient as an actual fixed computer. This means that such an environment has also to face CPU-related constraints, and it is a difficult, but interesting challenge to propose some solution which may improve or at least not degrade the performance of a given service and which warrants the user's privacy. For sensor networks, energy is an additional constraint which is an important factor. In fact, many efficient solutions can be found in the literature but they rarely treat simultaneously privacy in a very constraining context.

Traditionally, strong security and communication service are complementary; it is very often a question of trade-off. A strong security is time consuming and needs a large resource which can alter the quality of the services.

3 The properties of Reed–Solomon codes

Reed–Solomon codes. Let $\mathbf{F}_q$ be the finite field of q elements. Let $x_1, \ldots, x_n$ be n distinct elements of $\mathbf{F}_q$. We denote by $ev : \mathbf{F}_q[X] \to \mathbf{F}_q^n$, the evaluation function

$$ev : p(X) \mapsto (p(x_1), \ldots, p(x_n)),$$

where $\mathbf{F}_q[X]$ is the ring of the univariate polynomials over $\mathbf{F}_q$. We denote $RS_q(k, n)$ the Reed–Solomon code of dimension k and length n over $\mathbf{F}_q$. By definition

$$RS_q(k, n) = \{ev(f);\ f \in \mathbf{F}_q[X]; \deg f < k\}.$$

The minimal distance of this code is given by $n-k+1$, which guaranties the unicity of the decoding problem up to $n-k+1$ errors. We want to construct a secure protocol which lies on the difficulty of the Reed–Solomon decoding problem. The Decoding Problem of Reed–Solomon code (**RSD**) is defined by the following: *Given a $RS_q(k, n)$ code, ω an integer and a word $y \in \mathbf{F}_q^n$, find all codewords in $RS_q(k, n)$ at distance less than ω of y.*

Then we recall the main result concerning the complexity theory related to the **PR** problem (see in [37]).

Polynomial Reconstruction problem (PR): *Given n, k, t and $(x_i, y_i)_{i=1,\cdots,n}$ with distinct x_i's, find all polynomials $p(X)$ such that $\deg(p) < k$ and $p(x_i) = y_i$ for at least t values of the index i.* Therefore we have **PR = RSD**. It is known that **PR** is polynomial in n, k if $t > \sqrt{kn}$ (see [18]). It is also polynomial if $t = \sqrt{kn}$ with a complexity in $\mathcal{O}(n^{15})$ [18]. To understand the difficulty to solve the general problem of **PR**, we describe the following problem: the **Poly Agree Problem (PA)**. **PA**: *Given n, k, t and a set of pairs $P = \{(x_1, y_1), \cdots, (x_n, y_n)\}$, $x_i, y_i \in \mathbf{F}_q$, does there exist a degree k polynomial $p(X)$ such that $p(x_i) = y_i$ for at least t different i's?* It is important to note that for this problem, the x_i's are not required to be different, so this problem is not equivalent to the problem **PR**. The main complexity result is the following [19]: the problem **PA** is proved to be NP-hard so this result indicates that **PR** is a hard problem. The rest of this document focuses on the construction of a protocol based on the difficulty to reconstruct a polynomial (PR). We will describe this protocol in the following sections.

4 Multipath routing, path set optimization and security problems

A recent and popular idea consists in encoding k byte information with a polynomial of degree $k-1$ over a finite field $\mathbf{F}_q$, using a Lagrange interpolation to reconstruct the broadcasted information (see [33]). This is equivalent to encode the information with a Reed–Solomon code $RS_q[k-1, k]$ of length k and dimension k. If an attacker wants to reconstruct the information in a wired context, he needs to corrupt several nodes, which turns to be unpractical. Unfortunately, the attacker does not need to corrupt the totality of the nodes in the wireless context, but he can simply intercept the transmissions between nodes, and in particular it is not so difficult to intercept all transmission intended to a particular user if the node address is not protected. Hence, encoding with a Reed–Solomon code and splitting the information is not enough to perturb an attacker in a wireless context. In most cases, a proof of security can be obtained by assuming that the attacker has only access to a small fraction of the transmission as for instance into [32], in the context of sensor networks. The SPREAD protocol ([33]) aims at building some maximally disjoint multiple routes. Unfortunately in many cases, these disjoint paths may not exist, nevertheless, it is known that keeping the multipath method ensures an improvement of the QoS. Also, we would like to introduce the notion of threshold cryptography. If there is single path between the sender and the receiver like in Fig. 1, the problem of security cannot be solved without encrypting. We can use in these particular cases a fast symmetric cryptosystem. The Threshold depends on the desired security level a user wants. For instance we use encryption if the number of disjoint paths does not exceed D.

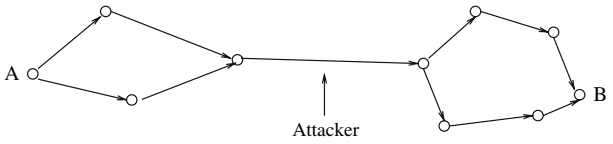


Fig. 1 No disjoint paths between A and B

One of the advantages of the mobile Ad Hoc networks is the ability to cover a large zone without a strong capacity of emission and reception. Therefore insuring the security of communication between users A and B may not be realized by using several disjoint paths as shown in Fig. 2.

In this case information must be encrypted. However we would like to keep the multipath method to insure a better QoS. For anonymity, the problem is similar, if the network is weakly dense and if only two users communicate together: a simple traffic analysis can break their anonymity. In fact in the trivial example of a network with two nodes, the anonymity has no sense, we only obtain the confidentiality

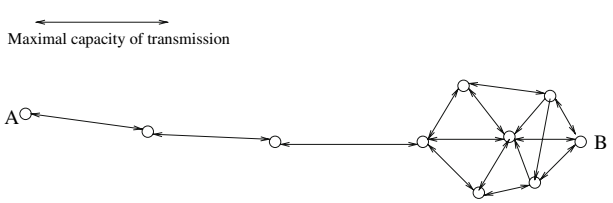
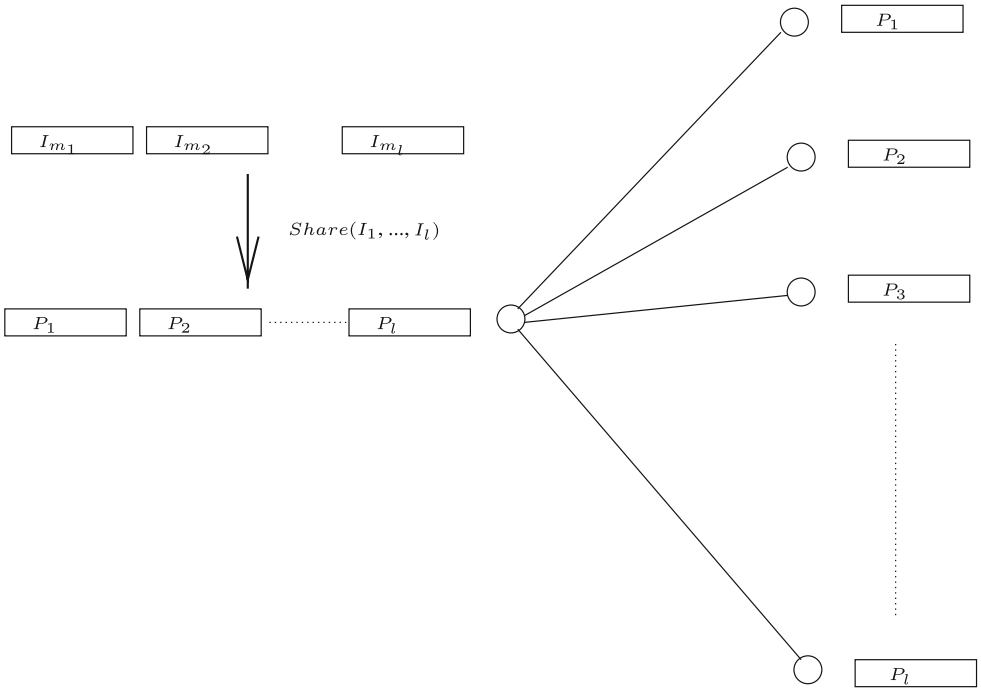


Fig. 2 No disjoint paths between A and B

Fig. 3 Splitting the message



of the transmitted information. Hence, assuming that the network has a reasonable number of nodes, then it is possible to insure partially anonymity. We propose in this article to use a method based on a fast public key encryption, a dummy traffic and a simple information which aims at transforming the plain information into an encoded information. We develop these ideas in the following sections. We give a global solution which under some assumptions works in any network configuration.

4.1 Adding QoS and security

Our idea is an extension of the SPREAD protocol (see [33]). The difference comes from the fact that our solution works in a noisy channel. The transmission can be corrupted by some bit error or burst error.

Reed–Solomon encoding We propose to encode the information with a Reed–Solomon code $RS_q[n, k]$ defined over $\mathbb{F}_q$ where q is usually equal to 2^8, of length $n < q$, dimension k and minimal distance $n - k + 1$. The generator matrix of this code is chosen to be non-systematic in order to make the encoded message not readable at once. We can correct with this code up to $n - \sqrt{kn}$ errors with a complexity of order $\mathcal{O}(n \log^2(n))$ byte operations (see [18]). Now we assume that in average there are $n\delta$ byte errors on a n byte information that we want to transmit through l paths. We propose to add a random error $E = (E_1, \ldots, E_n)$ of Hamming weight $W = n - \sqrt{kn} - n\delta$ for $W \geq 0$. According to our principle, the packet that we have to send has a fixed size proportional to n. Then each path can support $n\delta$ byte errors on each encoded message of length n. We call *Share* the function which adds some random byte error and sequences the transmitted packets. This function is described in Sect. 4.1. To encode an n bytes message m requires the following steps. First we encode m with a Reed–Solomon $RS_q[n, k]$ code that gives a n length vector $R = (R_1, \ldots, R_n)$, then we add byte per byte the random error E to this vector. We call this vector $I = (I_1, \ldots, I_n)$. This vector will be split in l shares $(I_1, \ldots, I_l)$ as shown in Fig. 3. The above procedure is simplified because many control messages should be added in order to insure the coherence and to synchronize the data which is a telecommunication engineering task.

$$(m_1, \ldots, m_k) \overset{RS_q[n,k]}{\longrightarrow} (R_1, \ldots, R_n),$$

and with an error E of weight W:

$$(R_1, \ldots, R_n) \longrightarrow (R_1 \oplus E_1, \ldots, R_n \oplus E_n) = I_m,$$

In the context of QoS, there is obviously a gain on error transmission. We also have an control on error coming from the channel and we improve the bandwidth requirement if the support has several antennas (e.g., laptop, PDA,...). In the case of a single antenna, a discussion is required and we give some simulation arguments below. Now from a security point of view, if an attacker wants to reconstruct a message, he has to intercept in average all the message shares P_j because each eventually corrupted packets P_j, $j \in [1, \ldots, l]$ of Fig. 3 contains only a fraction of the shares I_j, since the reconstruction problem is very difficult as shown in the previous section. To decode, the number of errors and erasures cannot be greater than the threshold value $n - \sqrt{kn}$ which corresponds

to the limit for which the decoding algorithm complexity is polynomial. Now we describe the function "Share" of Fig. 3.

How to share and split the message? First, the plain message is divided into k byte blocs $m_1, \ldots, m_k$ where k is the dimension of the chosen Reed–Solomon code $RS_q[n, k]$. Then each k byte bloc $m_1, \ldots, m_k$ is encoded with this Reed–Solomon code into a n byte bloc $R_1, \ldots, R_n$. Then an n byte error is added to the bloc $R_1, \ldots, R_n$ to get the message I_{m_j}, then setting $n = lp$, and partitioning I_{m_j} into l parts $I_{m_j}^{(s)}$, $s \in [1, \ldots, l]$ of p bytes:

$$I_{m_j} = \boxed{I_j^{(1)}}\boxed{I_j^{(2)}}\boxed{\cdots}\boxed{I_j^{(l)}} \; j \in [1, \ldots, l]$$

then a latency of l codewords m_j allows to construct l n-byte word P_i as follows

$$P_i = \boxed{I_1^{(i)}}\boxed{I_2^{(i)}}\boxed{\cdots}\boxed{I_l^{(i)}} \; i \in [1, \ldots, l]$$

and P_i corresponds to Fig. 3.

To get an effective solution, each unitary byte word P_j has to be concatenated with a control message. The security consequences are the following: in average, any fraction of potentially intercepted message is not exploitable for the reasons given before. This fraction of information is not directly readable since the generator matrix of the Reed–Solomon code was chosen to be non-systematic and we can choose $p < k$ in order to make it unreadable (even by inversion) even for non-noisy block of k bytes of P_j. We note that this technique is resilient toward malware intrusion. In fact if an intermediate node is corrupted, the attacker could introduce a malware in an intermediate node between the source and the destination, but from the point of view of a Reed–Solomon decoder, this malware is considered as noise and it will be corrected. So if an attacker wants to attack the network, he has to communicate with the destination, which means that he owns the private key of a node which belongs to the network, which is a strong hypothesis. It is easier for the attacker to force the communication to go through his node, but as we have seen, this method is not efficient when realizing a malware intrusion. At this point, our solution has still a drawback. In wireless context, it is not very difficult to intercept all transmission. Thus an attacker can decode the message as well as the receiver since the message and the control messages are not encrypted. We are going to repair this weakness in the following section by using a fast public key cryptosystem.

How to protect transmission? We use a fast public key cryptosystem. This layer is implemented above the layer containing the splitting algorithm.

The main idea is to encrypt the control message and the address node with a public key cryptosystem $Y = F(K_{pub}, X)$, where K_{pub} is the public key, X the plain message and Y the encrypted message. The inverse of $F(K_{pub}, X)$ is $F^{-1}(K_{priv}, X)$, where K_{priv} is the private key, i.e., $F^{-1}(K_{priv}, F(K_{pub}, X)) = X$. Usually public key cryptosystem are relatively slow, but the performance can be improved with a good hardware implementation. For instance for an elliptic curve, an encryption costs less than 3 ms (see [38]) with a FPGA implementation. In 1978, McEliece proposed a fast public key cryptosystem [36]. There is no known attack against this cryptosystem and within this cryptosystem, encryption is faster than RSA, Elliptic Curves and ElGamal cryptosystems.

Each node has its private key, and has the knowledge of the other user public key. We assume that the node A wants to send a message m to the node B through the path $\widehat{ACB}$. Let N_j be the address of node j for $j \in \{A, B, C\}$. As the number of nodes is not too large the address is concatenated with a strong random string. A sends to C the vector $(m, Y = F(K_{pub}(C), (N_B|random)))$. C is the only node that can recover B address N_B by decrypting this vector $(m, F^{-1}(K_{priv}(C), Y)) = (m, (N_B|random))$. Finally C can send m to B. We assume that in our protocol a receiver cannot determine the sender address (Contrary to IP protocol). With this method, the receiver C always knows the final destination B of the message, but he cannot determine who has sent this information, so, even if the node C is a foe, it cannot break anonymity. Even if C has access to a fraction of the message, he cannot reconstruct the complete information.

The control message (header) has to be ciphered using symmetric cryptosystem $Ciph(K_s, X)$ (like AES). A secret key K_s can be encrypted with the public key of the destination, and sent by the source. With this method, the intermediate nodes do not have access to the control message.

Notation: we denote m the message, Cm the control messages, K_s the secret key, N_d, the node address of the destination and rd, rd' some random bit strings. Therefore the encryption task has the following steps:

(1) Source A composes the string $(m|Cm|K_s|N_d)$, where the symbol | indicates a concatenation. Then source A sends to node C:

$$(m|Ciph(K_s, Cm)|F(K_{pub}(B), K_s|rd)|Ad(C)),$$

with

$$Ad(C) = F(K_{pub}(C), N_d|rd'),$$

where B is the destination and rd' is a random bit string.

(2) Destination node B receives

$$(m|Ciph(K_s, Cm)|F(K_{pub}(B), K_s|rd))$$

and computes

$$K_s = F^{-1}(K_{priv}(B), F(K_{pub}(B), K_s|rd)),$$

and

$$Cm = Ciph^{-1}(K_s, Ciph(K_s, Cm))$$

in order to get (m, Cm). Now an attacker has to recover the order of the received string in order to get eventually noisy codeword of Reed–Solomon code. This task is difficult because he does not know the control message of intermediate nodes.

If F represents the RSA cryptosystem and $Ciph$ is AES, we need several blocks to encrypt with Ciph because the block length of a symmetric cryptosystem is smaller than the bloc length of an asymmetric cryptosystem. If F is the 160 bit DCC (see [31]) and if $Ciph$ is the 256 bit Rijndael (see [13]), we have to append a random padding to the plaintext to get several blocks that are encrypted by the function F. The padding is compulsory because we do not want the same key to have the same plaintext to have a unique ciphered value.

We must also include a public and private key renewal mechanism. Such mechanism exists (see [4]) but is out of the scope of this article.

We have minimized the number of symmetric encryptions by only ciphering control message to reduce latency. However, if a user wants a stronger security, he can cipher the entire block plaintext as well. The QoS is lowered since the length of the block plaintext is considerably larger than the length of the control message. One bit of error in the transmission compromises immediately the deciphering of a block. Therefore a strong error-correcting codes should be considered.

To conclude, we give a good data protection and give a solution to solve the anonymity problem. Nevertheless there are only two active nodes in a network, a simple passive analysis of the traffic can break the anonymity of the two nodes. We propose to solve this problem in the following section in using the well known technique of "dummy-traffic".

How to solve the anonymity problem? If an attacker can intercept any transmission network, he can distinguish who is communicating with whom by using a simple traffic analysis. We propose to add some dummy traffic in the network: it improves anonymity and may provide unobservability. Real traffic should be treated before dummy traffic. For more details on dummy-traffic, see [12,24].

How to improve performance? We improve performance by splitting information and simulation. We assume that we have a strong routing protocol (like OLSR) that can give with some minor transformation the list of best paths in term of flow, noise, length of the path, number of intersections between source and destination. This protocol is able to attribute a value (less than one and positive) in any direction like in Fig. 4, directly proportional to these last criterion.

In this part, our goal is to show that our multipath strategy does not affect the quality of the service despite the fact that it increases security. It is not easy to give a proof that the multipath strategy gives better result than choosing the best route when supports have a single antenna, but we can give strong arguments. First if the noise is not uniformly distributed, then

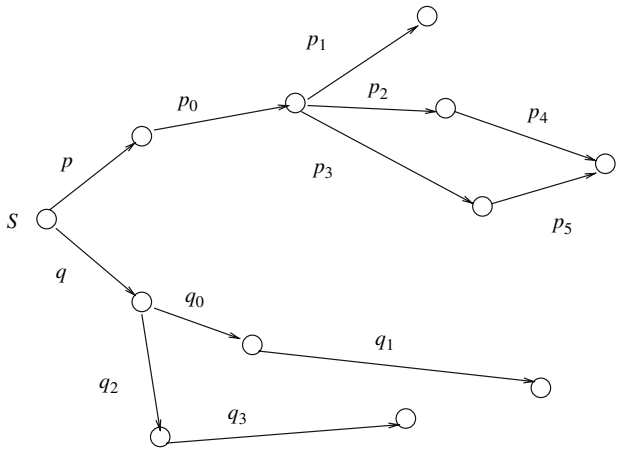


Fig. 4 Strategy of splitting

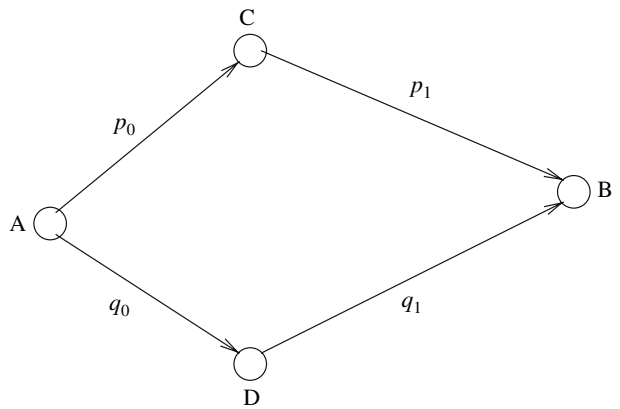


Fig. 5 Splitting performances

it is clear that by splitting, we increase the possibility to send decodable information. By construction, each value p_i or q_i depends on the quality of the next path. Then we can study this problem locally by considering Fig. 5.

We have affected at each edge a value that corresponds to the flow of the link. The flow p of the path $\widehat{ACB}$ is equal to $p = \min(p_0, p_1)$ and similarly, the flow q of the path $\widehat{ADB}$ is equal to $q = \min(q_0, q_1)$. Assume that the process which consists in sending alternatively some packets to node C and D is negligible compared to time for information to go from A to C or D. This is a reasonable hypothesis. If we consider that the different edges represent many node connections and by example if the flow is relatively slow, which is the case for wireless communication. Then the global flow of information is given by $p + q$, by counting the quantity of information at the nodes C and D. The unknown is the global flow between A and B since there is a conflict between the nodes C and D, each of them wanting to have access to node B. We propose to use a strong software that enables to simulate the conditions of the wireless network communication. We want to compare the flow that corresponds to use the

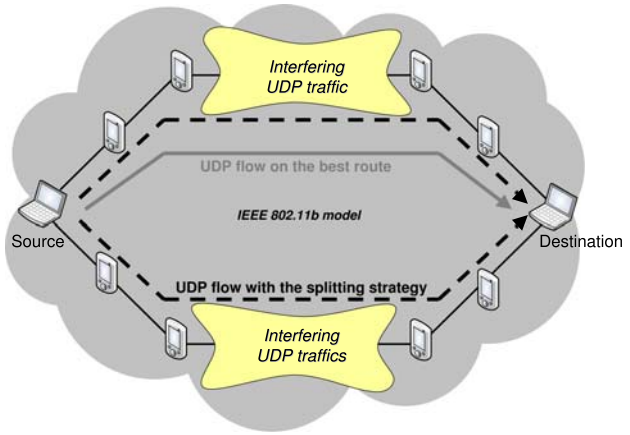


Fig. 6 Condition of the experience

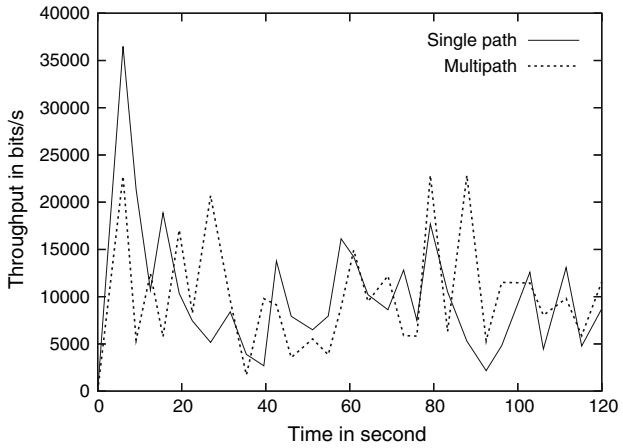


Fig. 7 OMNET++ result

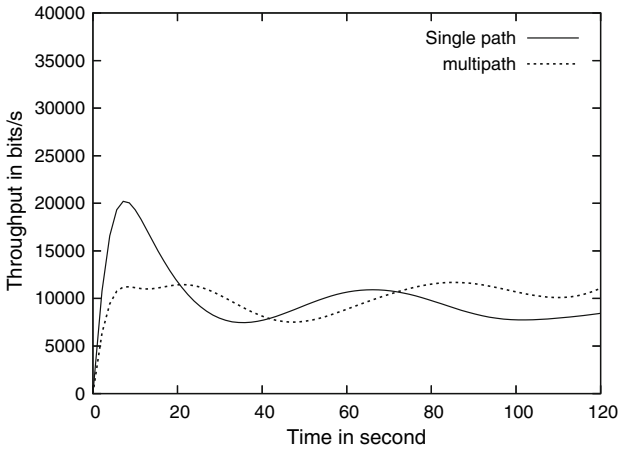


Fig. 8 Result with a smooth Bezier curve

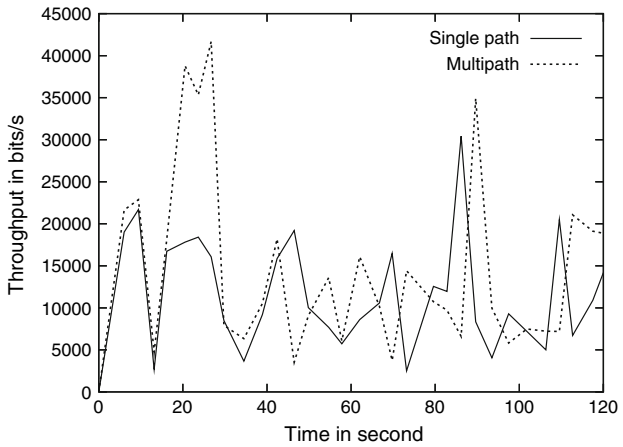


Fig. 9 OMNET++ result

best route between AB and the flow that consists in sending alternatively data through $\widehat{ACB}$ and $\widehat{ADB}$.

In order to study this issue, we implemented a model over OMNET++ (see [39]), an open-source discrete event simulation environment. The simulations rely on an INET framework (see [22]) libraries related to IEEE 802.11b. Figure 6 illustrates the scenario used to simulate our protocol. In this scenario, a UDP (unique direction protocol) connection is initiated by a source node toward the destination. Interference UDP traffic has been injected on different links composing the routes to impact on their global capabilities/quality.

First, a UDP flow is introduced between the source and the destination nodes following the best route (i.e., the less impacted one). Then, the UDP flow is split with respect to our strategy between two routes. Our strategy is to send data on each path alternatively. The simulation measurements consisted in compraing of the end-to-end throughput observed between the source and the destination nodes. We collect several measurements. It appears that the curves given by the OMNET++ software does not give a precise view of the traffic behaviour. Therefore we introduce for our simulation a Bezier approximation curve.

For Fig. 7, the average flow with the multipath strategy gives better results than a simple UDP connection: 11,162 bits/s against 10,284 with a single path (Fig. 8). For the Fig. 9, the average flow with the multipath strategy gives better results than a simple UDP connection: 14,181 bits/s against 11,391 with a single path (Figs. 10, 11). Only in Fig. 12, we observe an average flow with the multipath strategy that does not give better results: 12,270 bits/s against 12,985 with a single path.

The results of our simulations are rather good since we observe some gain in using the multipath strategy. So, this technique increases security and do not reduce the QoS, at least locally on the simple topology that we have constructed in Fig. 6. We show in Fig. 13 that in fact our strategy of splitting is recursive. We always have locally a gain. We show a global gain in a layer network using a recursive proof.

Of course the number of splits has to be bounded, and this bound has to be fixed in function of the quality of the support

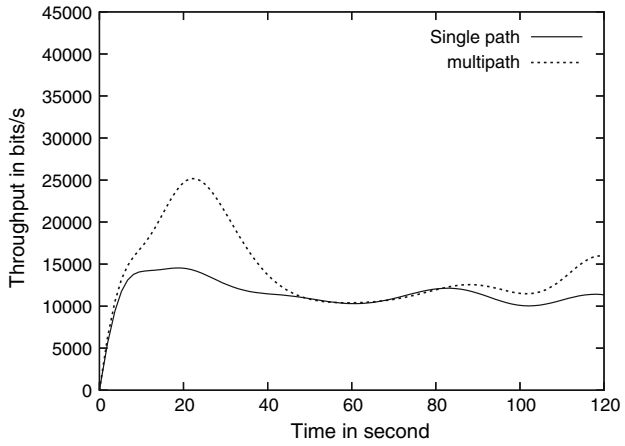


Fig. 10 Result with a smooth Bezier curve

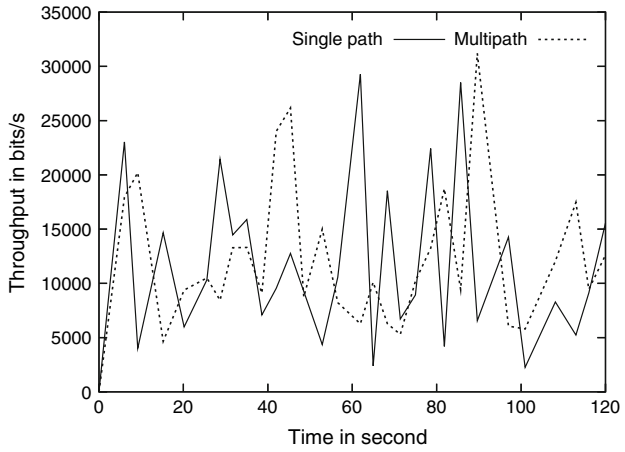


Fig. 11 OMNET++ result

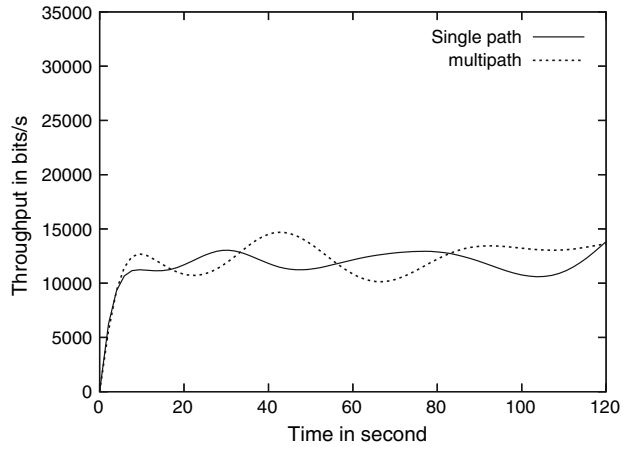


Fig. 12 Result with a smooth Bezier curve

(PDA, Laptop,. . .) and the transmission. A problem occurs when it is not possible to split as in Fig. 2 or when there are few disjoint paths. Then we introduce the notion of threshold cryptography in the following section.

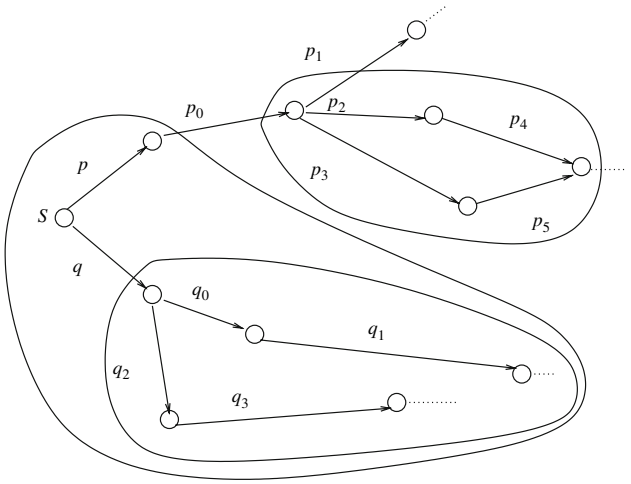


Fig. 13 Local strategy

How to resist to a malware intrusion? As explained in [25], Ad Hoc networks are more vulnerable than wired networks. Our protocol is not robust against active attacks which consists to alter the transmission, but such attack is not furtive and can be detected. In the same manner our solution probably does not resist against denial of services because the nature of Ad Hoc network makes it very sensitive to such attack. Another attack consists in corrupting a node by taking his identity and by sending a message that contains a malware. We can predict that only a short time is necessary to affect all nodes.

Corrupting a node and taking his identity is certainly not easy in practice. Our goal is to resist against furtive attacks: it consists in inserting a malware in the transmission. An active and furtive attack consists in playing the role of a node in the network and to be seen as node which belongs to the network. This corrupted node can intercept the transmission which is directed through itself to another node. When it intercepts the transmission he can encapsulate a malware in the data. We want to be robust against such attack.

The main routing protocols, OLSR, AODV, DSR, GSR, CGSR, DSDV, SPREAD are vulnerable to the described attacks (see [25]). Recently, an interesting solution was proposed in [25]. The authors of [25] propose a distributed architecture of IDS that can detect in real time many viral attacks. This technique is not in contradiction with our solution, and can be added in our protocol. This technique increases the security of our protocol.

Our protocol is resistant to the later attacks because the data which transits by this node (introduced in the Ad Hoc network) corresponds to a fraction of corrupted positions that belong to Reed–Solomon codeword. Thus the malware or furtive data is seen as an error transmission and is corrected by the decoder of the final receiver. When the topology of the Ad Hoc network does not allow a multipath strategy, there is

no important drawback because we have seen that the data are completely encrypted. This is our notion of threshold cryptography which is more detailed in the following section.

To conclude this part, our protocol is the only protocol with SPREAD that propose a splitting of the data for Ad Hoc networks, and compared to SPREAD, we have the advantage to resist against error transmission. Therefore, our protocol is more robust against the attack that we focus on. Furthermore our protocol can be improved by adding the solution of distributed IDS of [25]. We cannot definitively solve the malware attack problem because it is a very difficult problem (see [15,16]). Vulnerabilities could come from users, and it is difficult to control each user. We claim that our protocol is more robust than the previous protocols such as OLSR, AODV, DSR, GSR, CGSR, DSDV, SPREAD, and that it should be improved in further work.

How to warrant security with a changing topology network. We have seen in Figs. 1 and 2 that cases exist in which there are not enough disjoint paths. Then we introduce the following concept: if there are not more than l disjoint paths, then we cipher the information with a symmetric cryptosystem like the AES. The sender chooses a secret key and communicates in using an asymmetric cryptography. We predict that in average most communications will not require a complete encryption. The transmission process should stay relatively simple.

5 Conclusion

We have designed a method which brings security and privacy to network without losing QoS. First, we have generalized the approach of [33] and extended it to many complicated situations that could occur in an Ad Hoc network. We have used the software OMNET++ to give some results of simulation. These simulations validate our approach. We have shown in Sect. 4.1 that our solution resists against passive attacks, even if a reasonable fraction of nodes is corrupted by using a multipath strategy and asymmetric cryptography.

In our solution it is difficult to separate information anonymity from information protection. If the node addresse is not hidden, then anonymity is not insured. The dummy traffic used to insure anonymity is also useful to protect information as to reconstruct complete information, an attacker has to separate the useful information from the false information. We use the fact that we communicate through a network to insure the information confidentiality with a low computation cost.

We have shown in Sect. 3 that reconstructing a strong noisy message which is encoded with a Reed–Solomon code is a very difficult problem. We have also considered cases where a weakness could appear (few disjoint paths, few splits) and we introduce the notion of threshold cryptography. We have considered the case of malware attacks which could be a serious threat in Ad Hoc network. This point surely deserves further work. We have seen that using an existing solution (see [25]), our protocol is more resistant against malware attack than previous protocols. Our method is also very flexible since it is possible to cipher the complete message in order to get a higher level of confidentiality.

Of course the request for a route establishment, and the link quality message have to be delivered safely since they can leak information about the transmission. Nevertheless it should be judicious to consider different strategies according to the kind of data that we have to transmit. It appears that the required QoS for many applications can be very different. If we want to send voice through the network, we have to consider a real time transmission. We suggest for voice transmission to suppress any request when data does not find a route to have access to the receiver. Very often in transmission theory, there is a loss of packets when there are obstacles like a wall, a bridge... We can accept for voice some loss of packets. The most important for the voice and video transmission is the resynchronization of the data. For this purpose, a strong method should be used and one which is compliant with noisy channel. Such methods are currently used for the submarine and are based on periodic sequences that are introduced into the data. For a text file, it is simply not possible, we could loose information which prevents the document from opening. We have to consider a solution with secured requests and thus we have to accept more latency. Information concerning the kind of data to be transmitted could be included inside the control message. These considerations could be developed in a further work.

In this article, we have defined a strong and secured protocol for mobile Ad Hoc network and more generally for networks with strong constraints.

Acknowledgments We gratefully thank Olivier Orciere and Stephane Rouseau for their very helpful contribution to improve the document.

References

1. Boukerche, A., El-Khatib, K., Xu, L., Korba, L.: SDAR: A secure distributed anonymous routing protocol for wireless and mobile Ad Hoc networks. In: 29th IEEE International Conference on Local Computer Networks (LCN'04), pp. 618–624, November 2004
2. Balfanz, D., Durfee, G., Shankar, N., Smetters, D.K., Staddon, J., Wong, H.-C.: Secret handshakes from pairing-based key agreements. In: IEEE Symposium on Security and Privacy, pp. 180–196 (2003)
3. Berthold, O., Federrath, H., Kopsell, S.: Web MIXes: A system for anonymous and unobservable Internet access. In: Federrath, H. (Eds.) DIAU'00, Lecture Notes in Computer Science, vol. 2009, pp. 115–129 (2000)
4. Bhaskar, R., Augot, D., Issarny, V., Sacchetti, D.: An efficient group key agreement protocol for ad hoc networks. IEEE Workshop on Trust, Security and Privacy in Ubiquitous Computing (Affiliated with WoWMoM 2005), Taormina, Italy, 12–16 June 2005

5. Chaum, D.L.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM **24**(2), 84–88 (1981)
6. Clarke, I., Sandberg, O., Wiley, B., Hong, T.W.: Freenet: A distributed anonymous information storage and retrieval system. Lecture Notes in Computer Science (2001)
7. Castelluccia, C., Mutaf, P.: Hash-based dynamic source routing. In: IFIP Networking, LNCS, vol. 3042, pp. 101–223 (2004)
8. Clausen, T., Jacquet, P.: Optimized link state routing protocol (OLSR). IETF, Request For Comment 3626, October 2003
9. Chaum, D.: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Commun. ACM **24**(2), February 1981
10. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. In: Proceedings of 13th Usenix Security Simposyum, August 2004
11. Dingledine, R., Mathewson, N., Syverson, P.: TOR: The second-generation onion router (2004)
12. Diaz, C., Preneel, B.: Reasoning about the anonymity provided by pool mixes that generate dummy traffic. In: Fridrich, J. (Ed.) Information Hiding. LNCS, vol. 3200, pp. 309–325. Springer, Heidelberg (2004)
13. Daemen, J., Rijmen, V.: The block cipher Rijndael. CARDIS 1998. LNCS, vol. 1820, pp. 247–256 (2000)
14. El-Khatib, K., Korba, L., Song, R., Yee, G.: Secure dynamic distributed routing algorithm for Ad Hoc wireless networks. In: International Conference on Parallel Processing Workshops (ICPPW'03) (2003)
15. Filiol, E.: Computer Viruses: from Theory to Applications. IRIS International Series, Springer, France. ISBN 978-2-287-23939-7 (2005)
16. Filiol, E.: Techniques Virales Avancées. IRIS International Series, Springer, France. ISBN 978-2-287-33887-8 (2007) (An English translation is pending)
17. Freedman, M.J., Morris, R.: Tarzan: a peer-to-peer anonymizing network layer. In: Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002) (2002)
18. Guruswami, V., Sudan, M.: Improved decoding of Reed–Solomon and algebraic-geometry codes. IEEE Trans. Inform. Theory **45**, 1757–1767 (1999)
19. Goldreich, O., Rubinfeld, R., Sudan, M.: Learning polynomials with queries: the highly noisy case. SIAM J. Discrete Math. **13**(4), 535–570 (2000)
20. Hu, Y.-C., Johnson, D.B., Perrig, A.: SEAD: Secure efficient distance vector routing for mobile wireless Ad Hoc networks. In: Proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2002), p. 313, June 2002
21. Hu, Y.-C., Perrig, A., Johnson, D.B.: Ariadne: A secure on demand routing protocol for Ad Hoc networks. In: Proceedings of the Eighth Annual International Conference on Mobile Computing and Networking (MobiCom 2002), pp. 12–23 (2002)
22. INET framework for OMNET++. http://www.omnetpp.org/doc/INET/neddoc/index.html.
23. Johnson, D.B., Maltz, D.A., Hu, Y.-C.: The dynamic source routing protocol for mobile Ad Hoc networks (DSR). draft-ietf-manet-dsr-09.txt, April 2003
24. Jerichow, A.: Generalisation and security improvement of mixed-mediated anonymous communications. Ph.D. Thesis, Technischen Universitat Dresden (2000)
25. Jean-Marc, P.B.J.: Détection d'intrusions dans les réseaux Ad Hoc. SSTIC'03, 1er Symposium sur la Sécurité des Technologies de l'Information et de la Communication. Rennes, juin (2003)
26. Kong, J., Hong, X., Gerla, M., Sanadidi, M.Y.: Comparison: ASR is a variant of ANODR. Technical report, UCLA (2005)
27. Kong, J., Hong, X.: ANODR: ANonymous On demand routing with untraceable routes for mobile ad hoc networks. In: ACM MOBIHOC'03, pp. 291–302 (2003)
28. Kong, J., Hong, X., Gerla, M.: An anonymous on demand routing with untraceable routes for mobile Ad Hoc networks. Technical report CSD-TR030020, Department of Computer Science, UCLA (2003)
29. Kong, J.: Anonymous and untraceable communications in mobilewireless networks. Ph.D. Thesis, University of California, Los Angeles, June 2004
30. Kesdogan, D., Egner, J., Buschkes, R.: Stop-and-go MIXes providing probabilistic security in an open system. Second International Workshop on Information Hiding (IH'98), Lecture Notes in Computer Science, vol. 1525, pp. 83–98 (1998)
31. Koblitz, N.: Elliptic curve cryptosystems. Math. Comput. **48**, 203–209 (1987)
32. Luh, W., Kundur, D.: Distributed privacy for visual sensor networks via markov shares. In: Proceedings of 2nd IEEE Workshop on Dependability and Security in Sensor Networks and Systems. Columbia, MD, April 2006
33. Lou, W., Liu, W., Fang, Y.: SPREAD: Improving network security by multipath routing. IEEE Milcom'03. Boston, MA, October 2003
34. Lou, W., Liu, W., Fang, Y.: SPREAD: enhancing data confidentiality in mobile ad hoc networks. In: The 23rd Conference of the IEEE Communications Society (IEEE Infocom 2004), Hong-Kong, March 2004
35. Liu, J., Kong, J., Hong, X., Gerla, M.: Performance evaluation of anonymous routing protocols in mobile Ad Hoc networks. In: IEEE Wireless Communications and Networking Conference (WCNC), Las Vegas, NV, USA, 3–6 April 2006
36. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. DSN Prog. Rep., Jet Prop. Lab. California Inst. Technol., Pasadena, CA, pp. 114–116 (1978)
37. Naor, M., Pinkas, B.: Oblivious transfer and polynomial evaluation. In: Proceedings of the 31st Symposium on Theory of Computer Science (STOC), Atlanta, GA, pp. 245–254, 1–4 May 1999
38. Orlando, G., Paar, C.: A Scalable GF(p) Elliptic curve processor architecture for programmable hardware. In: Cryptographic Hardware and Embedded Systems CHES 2001: Third International Workshop, Paris, France, 14–16 May 2001
39. OMNET++. http://www.omnetpp.org/
40. Pfitzmann, A., Pfitzmann, B., Waidner, M.: ISDNMixes: untraceable communication with very small bandwidth overhead. In: GI/ITG Conference: Communication in Distributed Systems, pp. 451–463 (1991)
41. Perkins, C.E., Bhagwat, P.: Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers (1994)
42. Perkins, C., Belding-Royer, E., Das, S.: Ad Hoc on-demand distance vector (AODV) routing. RFC 3561, July 2003
43. Papadimitratos, P., Haas, Z.J.: Secure routing for mobile Ad Hoc networks. In: SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002). January 2002
44. Seys, S., Preneel, B.: ARM: Anonymous routing protocol for mobile Ad Hoc networks. In: 20th International Conference on Advanced Information Networking and Applications (AINA), Vienna, Austria, April 2006
45. Song, R., Korba, L., Yee, G.: AnonDSR: Efficient anonymous dynamic source routing for mobile Ad-Hoc networks. In: ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN) (2005)
46. Sanzgiri, K., Dahill, B., Levine, B.N., Shields, C., Belding-Royer, E.M.: A secure routing protocol for Ad Hoc networks. In: Proceedings of the 10th IEEE International Conference on Network Protocols (ICNP) (2002)
47. Sy, D., Chen, R., Bao, L.: ODAR: On-demand anonymous routing in Ad Hoc networks. Mobile Adhoc and Sensor Systems (MASS),

2006 IEEE International Conference, pp. 267–276, Vancouver, Canada, October 2006
48. Venkatraman, L., Agrawal, D.P.: Strategies for enhancing routing security in protocols for mobile Ad Hoc networks. In: Journal of Parallel and Distributed Computing, 63.2 (February 2003). Special issue on routing in mobile and wireless Ad Hoc networks, pp. 214–227 (2003). ISSN:0743-7315
49. Yang, H., Meng, X., Lu, S.: Self-organized network-layer security in mobile Ad Hoc network. In: Proceedings of the ACM Workshop on Wireless Security, pp. 11–20 (2002)
50. Yi, S., Naldurg, P., Kravets, R.: Security-aware Ad Hoc routing protocol for wireless networks. In: The 6th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 2002) (2002)
51. Zhu, B., Wan, Z., Kankanhalli, M.S., Bao, F., Deng, R.H.: Anonymous secure routing in mobile Ad Hoc networks. In: 29th IEEE International Conference on Local Computer Networks (LCN'04), pp. 102108, November 2004
52. Zhang, Y., Liu, W., Lou, W.: Anonymous communications in mobile Ad Hoc networks. In: IEEE INFOCOM (2005)