2012 31st International Symposium on Reliable Distributed Systems

# Anonymous On-demand Routing and Secure Checking of Traffic Forwarding for Mobile Ad Hoc Networks

Rui Jiang
School of Information Science and Engineering
Southeast University
Nanjing, China
R.Jiang@seu.edu.cn

Yuan Xing
School of Information Science and Engineering
Southeast University
Nanjing, China

*Abstract*—**Anonymous communications in mobile ad hoc networks is an important and effective way against malicious traffic analysis. Various anonymous routing schemes have been proposed for MANETs. However, most works failed to resist the global tracker, and always ignored the reliability of data delivery. In this paper, a comprehensive anonymous communication protocol, called ARSC, is proposed. The ARSC consists of anonymous routing, which is based on identity-based encryption pseudonym and single-round onion, and secure checking of traffic forwarding in data transmission phase, to achieve strong route anonymity and improve reliability of packet delivery in the data transmission phase. From the security analysis, our protocol ARSC is more secure than other schemes such as ANODR, SDAR, AnonDSR, CAR and MASK. Moreover, simulation experiments show that the ARSC has better performance than any other onion-structured anonymous routing protocols.**

***Keywords-MANETs, on-demand routing, security, anonymous***

## I. Introduction

A Mobile Ad Hoc Network (MANET) is a self-organized and self-configuring network, which is more vulnerable to various attacks than conventional wired and wireless networks due to the absence of fixed infrastructure, the dynamic network topology, and the restricted energy resources. The traditional end-to-end security mechanisms can provide confidentiality for transferred data between communication nodes, but adversaries can easily overhear all messages 'flying in the air' without physically compromising nodes. In some security-sensitive scenarios, such as military ad hoc networks, the exposure of the locations of command centers or mobile high priority nodes will enable the adversaries to launch pinpoint attacks on them. Thus, rational anonymity communication protocols should be designed to restrict the collection of network information by eavesdropping in order to protect the privacy of nodes.

Anonymity for communication data means preventing adversaries from linking the communication message with the source or the destination, and making an anonymous route so that each intermediate forwarding node only knows its local route pseudonym and does not know who else is on the same anonymous route. The concept of anonymity could be summarized as entity anonymity, route anonymity, and topology anonymity [1]. From now on, various anonymous routing schemes have been proposed for MANETs. The ANODR [2] is an anonymous on-demand routing which conceals the network identifiers of communicating nodes using onion routing [3, 4]. But it does not describe how the source and destination establish a shared secret key. Furthermore, a global attacker can easily find the path only by tracking and comparing the transferred data packet since there is an invariant element in the communication data part. The SDAR [5] improves the trapdoor solution. However, the security hazard is all the intermediate nodes will be exposed to the destination node. More seriously, the attacker can easily delete the last part of the variable-size path information. In AnonDSR [6], the authors improve trapdoor and onion design in SDAR to increase the anonymity degree as well as lower the computational complexity. In article [7], the author limits the number of onion nodes on a route to improve the protocol's scalability for large-scale ad hoc networks.

Besides the onion-structure protocol, there are some protocols adopting novel method, including CAR[8], A3RP[9], MASK[10], PRISM[11], A-HIP[12]. However, these research works still failed to achieve strong route anonymity so that the global tracker could obtain the route information through tracking the data stream. Another Problem of these research works is the reliability of data delivery has been ignored.

Ad hoc networks are prone to self-interest and malicious behavior. For example, a misbehaving node on the detected path may simply refrain from participating in data forwarding; it may selectively drop packets to degrade performance. Some solutions against packet dropping have been proposed in [13-16]. But none of them is designed for anonymous system.

In this paper, we propose a novel anonymous communication protocol, called ARSC, to solve the above issues. The ARSC includes anonymous routing, which is based on identity-based encryption pseudonym and single-round onion routing, and secure checking of traffic forwarding based on hash chain. Our ARSC differs from previous anonymous communication systems for MANETs mainly in the following aspects. First of all, our protocol does not require extra regular neighboring authentications or pair key establishing procedure; instead, it integrates a suite of inter-operative authenticated key exchange mechanisms into the routing algorithm design. Second, by keeping the visible message dynamic in both the RREP packet and the data packet, the ARSC could achieve the strong route anonymity against the global tracker, which is always neglected by most other research works. Moreover, the ARSC adopts the concept of onion routing, and only uses it in the RREP phase. The single-round onion routing is more efficient than the traditional two-round onion routing schemes. Finally, the passive attack in the data transfer phase such as packet dropping could be discovered through intermediate node checking the hash value.

## II. System Assumptions and Initiation

### A. *Assumptions and Notations*

A Trust Authority (TA) is required to generate and issue public keys before a node takes part in the network. We assume that each node can efficiently compute cryptographic algorithms and further assume that a source node knows the constant public key of the destination node. Finally, while the network is running, the time clock can be maintained by each node with exceedingly low error rate to judge the timeliness of route request. Table I summarizes the main notations used in our protocol.

TABLE I. Notations used in the paper

| Notations | Meanings |
|---|---|
| rreq,rrep,dfwd | Used to represent the three packet types |
| pseudoi | Pseudonym of node Ni generated by TA |
| IDs, IDd | The real identities of the source NS and destination ND |
| < Pi, Si> | Constant private key and public key of node Ni generated by TA. Where $P_i = H(pseudo_i)$ , $S_i = \omega P_i$ |
| <PKi,SKi> | Dynamic public key and private key of Node Ni |
| <PKtemp, SKtemp> | The temporary public key and private key generated by destination |
| Ki,i+1 | The session key between two adjacent nodes |
| Ki | The temporary session key generated by node Ni |
| {}* | Encryption by PKi, PKtemp, Ki,i+1, or Ki |
| $\{\}_{SK_S}$ | Signature by the dynamic private key SKs of source NS |
| ri, ni | Random numbers generated by node Ni |
| RAND,r | Random number generated by the source |
| $CH_S^j$ | The j times hashing value generated by the source |
| Oi | The route pseudonym of node Ni ,and Oi=PKi |
| Te | The timestamp for the RREQ message |

### B. *Initiation of System*

TA Determines an additive group $G_1$ of prime order $q$ as a group of points on an elliptic curve over $F_q$, a multiplicative group $G_2$ of the same prime order $q$ of a finite field $F_{q^k}$ for $k \in Z_q^*$ where $Z_q^* = \{y \mid 1 \le y \le q-1\}$, a bilinear mapping $g: G_1 \times G_1 \to G_2$ satisfying the properties of cryptographic bilinear map, and collision resistant cryptographic hash functions $H$ and $H_0$, where $H: \{0,1\}^* \to G_1$ mapping from strings of arbitrary length to points in $G_1$ and $H_0: \{0,1\}^* \to \{0,1\}^\mu$ mapping from strings of arbitrary length to output of fixed length. TA also generate a system's secret $\omega \in Z_q^*$, where $Z_q^* = \{y \mid 1 \le y \le q-1\}$. Thus, $< G_1, G_2, g, H, H_0 >$ is the system public parameter issued to the system legal members.

When a legitimate node $N_i$ is allowed to join the network, TA assigns it a pseudonym *pseudo$_i$*. Then its constant public key is $P_i = H(pseudo_i)$, and its private key is defined as $S_i = \omega P_i = \omega H_1(pseudo_i)$. Each node stores the set $< P_i, S_i >$, and according to the Discrete Logarithm problem, we can know that no one could determine the system secret $\omega$.

Each node can generate its dynamic public key $PK_i$ by computing $PK_i = r_i P_i = r_i H(pseudo_i)$ which is also functioned as its dynamic route pseudonym $O_i$. $r_i$ is a random number selected by node $N_i$. And the corresponding dynamic private key is $SK_i = \omega PK_i = \omega r_i P_i = r_i S_i$. The set $< PK_i, SK_i >$ still holds the properties of cryptographic bilinear map.

According to the Computational Diffie-Hellman problem, the node $N_i$ and node $N_j$ could compute their session key $K_{ij}$ through exchanging their dynamic public keys $PK_i$ and $PK_j$, where

$$K_{ij} = g(SK_i, PK_j) = g(\omega PK_i, PK_j) = g(PK_i, PK_j)^\omega,$$

$$K_{ji} = g(PK_i, SK_j) = g(PK_i, \omega PK_j) = g(PK_i, PK_j)^\omega,$$

and $K_{ij} = K_{ji}$.

## III. Description of ARSC Protocol

### A. *Route request (RREQ) phase*

**Source node:** Whenever a source $N_S$ communicates to a destination $N_D$, if a valid route is not in its route cache, it launches route discovery request to $N_D$. First, $N_S$ generates two new dynamic public key pairs $< PK_S, SK_S >$ and $< PK_0, SK_0 >$, and generates a hash value $CH_S^m$ by hashing $r_S$ $m$ times ($m>T_e$). $N_S$ saves the hash chain values $CH_S^1$, $CH_S^2$, …, $CH_S^m$, then creates the following RREQ packet and broadcasts RREQ locally.

RREQ$_S$=$rreq, tag, PK_0, Vchain_S, Trd$

Where $tag \in Z_q^*$ is a sequence number, $Vchain_S = \{RAND\}_{K_S}$ which will be encrypted layer upon layer, *Trd* is a public key cryptographic trapdoor that defined as $Trd = \{tag \| ID_S \| ID_D \| \{T_e \| r \| PK_S \| CH_S^m\}_{S_S}\}_{P_D}$ . $N_S$ inserts a record {*tag*, $ID_D$, <$PK_S$, $SK_S$>, <$PK_0$, $SK_0$>, *RAND*, *r*} into its routing table.

**Intermediate node:** Once a RREQ message is received, each intermediate node $N_i$ seeks the identical *tag* in its routing table. If same *tag* exists in the table, the given packet is discarded. Otherwise, each node $N_i$ verifies the timestamp $T_e$, and tries to use its constant private key to decrypt the trapdoor *Trd*. If the top of the decrypted thing is different from the *tag*, it is not the target. It generates its own dynamic public key set, and replaces the last hop node's dynamic public key $PK_{i-1}$ with its own dynamic public key $PK_i$. After that, each node $N_i$ generates a temporary session key $K_i$ that will be shared among the source, destination and itself, and then replaces the

*Vchain*$_{i\text{-}1}$ with $Vchain_i = \{Vchain_{i-1}\}_{K_i}$ . The renewed RREQ message is

RREQ$_i$=$rreq, tag, PK_i, Vchain_i, Trd, T_e$

Each node $N_i$ records $<tag, PK_{i-1}, SK_i, K_i, Vchain_i>$ in its routing table and maintains it for a short period.

**Destination node:** When the RREQ packet reaches at the destination $N_D$, $N_D$ verifies the signature $\{T_e \| RAND \| PK_S \| CH_S^m\}_{S_S}$ . If the signature is true, similar to the previous nodes, $N_D$ generates the dynamic public key pair $<PK_n, SK_n>$, and computes the session key with the last hop node, for example node $N_j$, where $K_{Dj}$=$g(PK_j, SK_n)$=$g(PK_j, \omega PK_n)$= $g(PK_j, PK_n)^{\omega}$ . $N_D$ also generates another different dynamic public key pair $<PK_D, SK_D>$, and computes the session key with the source $K_{DS}$= $g(PK_S, SK_D)$=$g(PK_S, \omega PK_D)$= $g(PK_S, PK_D)^{\omega}$ .

## *B. Route Reply (RREP) Phase*

**Destination node:** The destination $N_D$ generates a temporary one-time public key pair $<PK_{temp}, SK_{temp}>$, and makes a set of association $<O_D, Y/N>$, in which $O_D = PK_D$ is the route pseudonym and the symbol *Y or N* indicates itself is or isn't the destination. $N_D$ constructs the RREP packet as follows and broadcasts it to the last hop node $N_j$.

$$\text{RREP}_D = rrep, PK_n, Vchain_j, \{tag \| PK_{temp} \| onion_D \| CH_S^m \| PK_D \| \{tag \| ID_S \| ID_D \| r \| SK_{temp}\}_{K_{DS}}\}_{K_{Dj}}$$

Where $onion_D = \{tag \| r\}_{PK_{temp}}$ , and *VChain*$_j$ is not altered.

Finally, $N_D$ includes records $\{tag, ID_S, K_{Dj}, K_{DS}, <PK_{temp}, SK_{temp}>, <O_D, Y/N>\}$ into its routing table.

**Intermediate node:** Once $N_i$ receives the above RREP packet, $N_i$ computes the session key with $N_{i+1}$ as $K_{i,i+1}$ = $g(SK_i, PK_{i+1})$ = $g(PK_i, PK_{i+1})^{\omega}$ =$K_{i+1,i}$. Hence, $N_i$ could decrypt the encrypted message part and check *tag* and *VChain*$_i$. If there exists a matched *tag* stored in the routing table and the verification of *VChain*$_i$ successes, $N_i$ gets *VChain*$_{i\text{-}1}$ from decrypting *VChain*$_i$ with the corresponding temporary key $K_i$. After that, $N_i$ adds $<O_i, N>$ and the destination's route pseudonym $O_{i+1}$ into the routing table (During the data forwarding phase, we call the dynamic public key $PK_i$ as route pseudonym $O_i$).

In addition, $N_i$ also could compute the session key with the last hop node which is $K_{i,\,i\text{-}1}$ = $g(SK_i, PK_{i\text{-}1})$ = $g(\omega PK_i, PK_{i\text{-}1})$ = $g(PK_i, PK_{i\text{-}1})^{\omega}$. Using this key, $N_i$ updates the RREP packet as follows and broadcasts it to the last hop node $N_{i\text{-}1}$.

$$\text{RREP}_i = rrep, PK_i, Vchain_{i-1}, \{tag \| PK_{temp} \| onion_i \| CH_S^m \| PK_D \| \{tag \| ID_S \| ID_D \| r \| SK_{temp}\}_{K_{DS}}\}_{K_{i,\,i-1}}$$

Where $onion_i = \{onion_{i+1} \| K_i\}_{PK_{temp}}$ .

Similarly, subsequent nodes successively execute the same operation until the *RREP* packet finally arrives at the source. Finally, the routing table of the intermediate node $N_i$ is as follows. The flag *cnt* is a counter that is enabled when the node first uses this route to transfer data.

| | | | | | | |
|---|---|---|---|---|---|---|
| $<O_i,N>$ | $O_{i+1}$ | $K_{i\text{-}1,i}$ | $K_{i,i+1}$ | $K_i$ | $CH_S^m$ | $cnt_i$ (=0) |

**Source node:** When the RREP packet reaches at the source node $N_S$, $N_S$ computes the session key $K_{S1}$ and $K_{SD}$, and decrypts *onion*$_i$ layer by layer with private key $SK_{temp}$. Hence, it obtains the temporary session keys $\{K_1, K_2, \ldots, K_i, \ldots K_j\}$ of all the intermediate nodes and stores them in the routing table. Finally, the routing table of the source is as follows:

| $ID_D$ | $O_1$ | $K_{S,D}$ | $K_{S,i}$ |
|---|---|---|---|
| $\{K_1, K_2, \ldots, K_i, \ldots K_j\}$ | $\{CH_S^1, \ldots, CH_S^m\}$ | $cnt_S$ (=0) | |

## *C. Data Forwarding Phase*

In our protocol, the final hash chain value generated by the source is distributed to each participating nodes during the RREQ phase. After the route discovery done, whenever the source wants to send a data packet to the destination, it randomly picks an intermediate node $N_i$ in advance as a traffic checking node. And every node on the path accumulates its corresponding counter $cnt_i$ once it correctly receives a data packet. When the source sends the *k*'th data packets to the same destination, $cnt_S$=*k* and the *k*'th data packet is as follows:

$$PACKET_i = dfwd, O_1, \{\{K_i \| CH_S^{m-k}\}_{K_i} \| \{K_i \| data\}_{K_{SD}}\}_{K_{S1}}$$

Each node receiving this packet tries to decrypt the encryption part $\{K_i \| CH_S^{m-k}\}_{K_i}$ using its own temporary session key. If the node could decrypt the encryption part correctly, it means the node is the designated traffic checking node. Then the traffic checking node fetches its counter $cnt_i$ from routing table, and hashes $CH_S^{m-k}$ $cnt_i$ times. If $H_0(CH_S^{m-k})^{cnt_i} \neq CH_S^m$ , then node $N_i$ creates a misbehavior report flooding to the source. If $H_0(CH_S^{m-k})^{cnt_i} = CH_S^m$ , then the data message is forwarded conventionally.

Finally, in order to prevent the designated traffic checking node being attacked and replaced by the malicious node, the destination node should check whether the encryption part $\{K_i \| data\}_{K_{SD}}$ is correctly associated with $\{K_i \| CH_S^{m-k}\}_{K_i}$ or not.

## IV. ANONYMITY AND SECURITY ANALYSIS

## *A. Anonymity Analysis*

**Entity anonymity:** Every participating node uses its pseudonym $P_i$ issued by TA and transforms it into a different pseudonym $PK_i$ every time when a new route request is launched. Hence, the source and destination nodes cannot recognize the real identities of the intermediate nodes. Moreover, the source and destination nodes even cannot associate any link of the intermediate nodes from multiple route discoveries. At the same time, the real identities of the source and destination nodes which are included in the RREQ message are encrypted by destination's public key, hence only the source and destination nodes can know each other.

**Route anonymity:** While the route discovery process runs, the source, intermediate, and destination nodes only have the pseudonyms of previous and next hop, and some verification information. This means even if the participate nodes have no idea about the path between the source and destination. However, there may be another case: the global adversaries can track the RREP packet or data packet in line with the invariant information of packets, so that they can determine a certain route. This case is always neglected by the previous works, but our protocol could prevent this case happening. In our protocol ARSC, besides the symbol *rrep*, the RREP packet includes terms $PK_i$, $Vchain_{i-1}$ and $\{*\}_{K_{i,i-1}}$, which keep changing at each hop. Thus, the global adversaries cannot derive a certain route by tracking because every RREP packet for different route discoveries cannot be distinguished. For the same reason, the data packets can resist tracking.

**Topology anonymity:** In ARSC, there is no clue about hop count of packets because the packets keep identical size for the route discovery. And also, the packet structure does not include any information to determine the network topology and the number of participating nodes, except that the source knows the hop number of the whole path.

## B. Security Analysis

TABLE II. PROPERTIES COMPARISON OF SEVERAL PROTOCOLS

| Properties | ANODR | SDAR | CAR | MASK | AnonDSR | ARSC |
|---|---|---|---|---|---|---|
| Entity anonymity | √ | | √ | | √ | √ |
| Route anonymity | | | | | √ | √ |
| Topology anonymity | √ | √ | √ | √ | √ | √ |
| Resist forgery attacks | √ | √ | | √ | √ | √ |
| Resist data packet dropping | | | | | | √ |

**Forgery attacks:** Our protocol provides end-to-end authentication through the source and destination verifying the signature and generating the session key $K_{SD}$. Any exterior adversary cannot impersonate as a valid node to negotiate session key with the legal node because the system secret $\omega$ is confidential.

**Data packet dropping:** In our protocol, the traffic checking node is designated by the source in a random and secret fashion. And the test hash value is also protected by the traffic checking node's temporary key. The verification of the hash value is to ensure that the data packet number counter $cnt_i$ of each node is always identical to the counter $cnt_S$ of the source, which is the actual quantity of sent packets. If any compromising or misbehaving intermediate nodes arbitrarily drop data packet, the counters of the subsequent nodes must have discrepancy with the right number. Therefore, the verification will fail laterly. Thus, the data packet dropping attack will be discovered by the next traffic checking node.

Compared with some other anonymous routing protocols, the ARSC has some advantages listed in table II. From table II, ARSC and AnonDSR are more secure than the others.

## V. PERFORMANCE SIMULATIONS

### A. Cryptographic Implementation

The ARSC combines symmetric key cryptographic algorithm, public key cryptographic algorithm, and hash function together. To have a fair comparison, in the simulation we unify the cryptographic mechanism, key and field size. We assume ARSC uses ECIES (256-bit key) as public key cryptographic algorithm while other protocols use RSA (2048-bit key). All the protocols use AES-CBC (256-bit key & block) as symmetric key encryption and MD5 (128 bit output) as a hash function.

Table III shows the processing time of different cryptographic operations in paper [17], which is measured on an Intel Core 2 1.83 GHz processor under Windows Vista in 32-bit mode. For public key cryptosystems, the table shows processing latency per operation. For symmetric key cryptosystems, it shows operation bit-rate.

TABLE III. PROCESSING TIME FOR VARIOUS CRYPTO SYSTEMS

| Cryptosystem/Hashing | Processing speed |
|---|---|
| ECIES encryption/verification | 4.21ms/operation |
| ECIES decryption/signature | 3.98ms/operation |
| RSA encryption/verification | 0.16ms/operation |
| RSA decryption/signature | 6.08ms/operation |
| ECDH key agreement | 2.82ms |
| DH key agreement | 3.84ms |
| AES-CBC | 80Mbps |
| MD5 | 255 Mbps |

### B. Simulation Environment and Metrics

The simulation is performed in omnet++[18]. The network coverage area is a 1200-meter×600-meter area, within which 100 mobile nodes initially uniformly placed. Each node has a radio power range of 250 meters. Each run executes 900 seconds of simulation time. The supported channel capacity is 2 Mbps. The IEEE 802.11 Distributed Coordination Function is used as the MAC layer protocol to model transmission, queuing, and propagation delays and to provide reliable communication at the data link level. The sources and destinations are chosen randomly with uniform probabilities. Random Way Point (RWP) model is used to simulate node mobility, and the pause time is 30s.The size of all data packets is set to 512 bytes and the interval time to send packets is 0.25 second.

We present two sets of simulations. One set is to show routing performance variation when nodes speed increases from 0 to 10m/s under secure environment. The performance metrics are network connectivity, route establishing latency and control message overhead. In the other set of simulation, the malicious adversaries are introduced into the network. The adversaries are randomly selected out from the network nodes, and they reject to transmit the data packet. The main metrics are connectivity which represents the effect of misbehavior on route establish phase, and the packet delivery ratio which represents the effect of misbehavior on data transmission phase.

**Connectivity:** this is the ratio of the successful route discovery requests to the total route requests sent by the source. This represents the success probability of the route establishment.

**Route establishing latency:** the overall route establishing latency including the queuing, transmission and packet handling delays of RREQ message and RREP message. The packet handling delays are mainly based on the protocols and the actual measurement of the cryptographic systems mentioned in section 6.1.

**Normalized control overhead:** the normalized control message overhead is the total control packets per data packet delivered in terms of bytes.

**Packet delivery ratio:** this refers to the ratio of data packets correctly received by the destination to the data packets actually sent by the source under the adverse network environment.

### *C. Simulation Results*

#### *1) In Secure network environment*

Fig.1 shows that ARSC and AnonDSR have approximately equal connectivity and the connectivity of both protocols a little bit varies as the speeds of the nodes increase.

Fig.2 illustrates the route establishing latency. Moreover, two presuppositions are made. First, overhead incurred in pre-configure phase or bootstrap phase, such as required for key generation and key distribution, is not counted in the evaluation. Second, since the ARSC protocol embodies the pairwise key exchange in the route discovery process, in order to keep fair comparison criteria, the security parameter establishment (SPE) protocol for AnonDSR protocol is also considered into the evaluation, i.e., the average overhead and latency will be calculated.

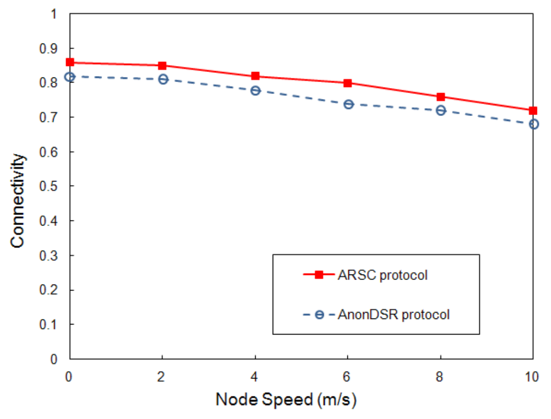

Figure 1. Connectivity in Secure Network

From Fig.2, route establishing latency of both protocols increases as the speeds increase, and that ARSC has a slightly lower latency than AnonDSR. The main reason is that the ARSC only includes single-round onion structure in the RREP phase, while the AnonDSR has two-round onion structure. The latency comes from the symmetric-structure onion of AnonDSR which requires the source node do *n* times more symmetric decryptions.

Fig.3 compares the normalized control overhead in terms of bytes. The normalized control bytes of both ARSC and AnonDSR are large because each packet needs carry a lot of keys, key negotiation materials and relevant verifiable information. The ARSC generates less normalized control bytes than AnonDSR. The result is expected, because AnonDSR has one more onion structure, the size of which is increasing along with the hops of returned RREP message. In addition, the periodically executed SPE protocol needs extra control bytes.

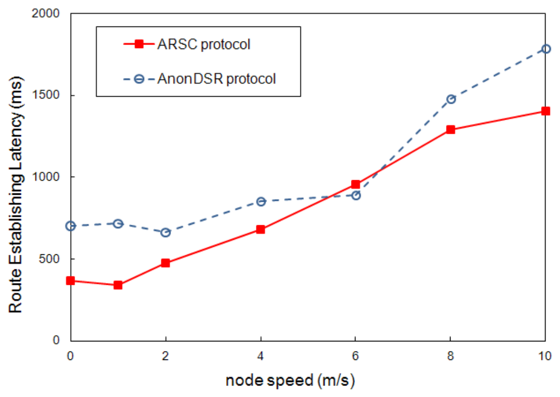

Figure 2. Route Establishing Latency

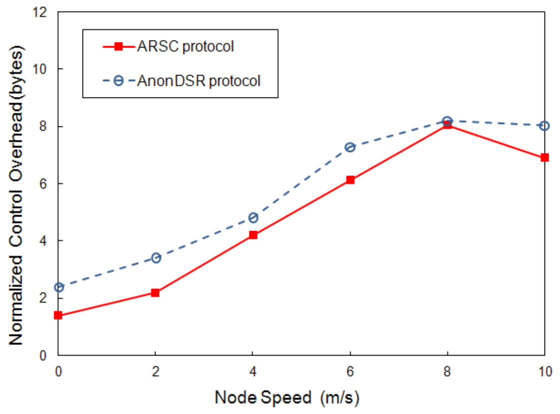

Figure 3. Normalized Control Overhead

#### *2) In adversary network environment*

Unlike the experiments described in the previous section, in this section, we measure the influence of malicious nodes on ARSC and AnonDSR. We involve malicious nodes in this set of experiments to illustrate the effect with the increase of their amount for the ARSC and AnonDSR protocols respectively.

Fig.4 shows that ARSC and AnonDSR have approximately equal connectivity under the adversary environment, and the connectivity of both protocols rapidly declines as the amount of malicious nodes increases.

Fig.5 compares the packet delivery ratio of ARSC scheme and AnonDSR scheme. Under the function of different percentage of misbehaving nodes that varied from 0 to 40%, the packet delivery ratio decreases as more nodes in the network misbehave. We can observe that ARSC could successfully deliver more data packets than AnonDSR under the same conditions. The fast degradation of the AnonDSR protocol comes no surprise, since AnonDSR does not provide any security mechanism against the data packets dropping, and the source node could detect a compromised route only when a link breakage is reported.

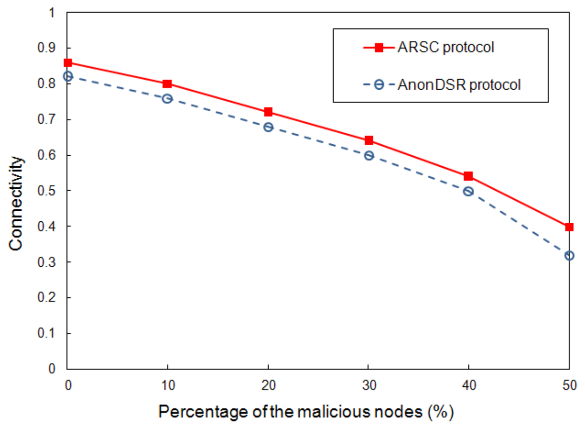

Figure 4. Connectivity in Adversary Network

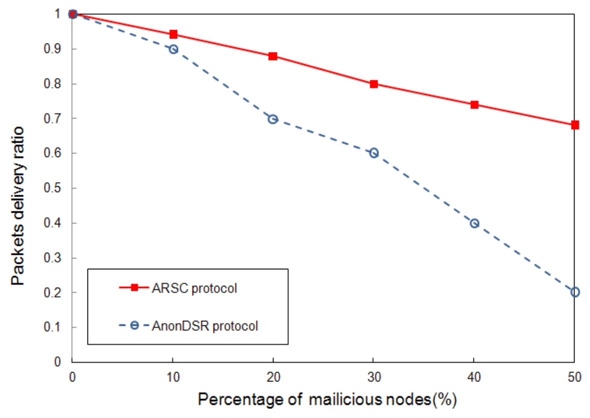

Figure 5. Packets Delivery Ratio

## VI. CONCLUSIONS

In this paper, we propose a comprehensive anonymous communication protocol, named ARSC, which includes anonymous routing phase and secure traffic forwarding phase. Besides achieving the strong anonymity properties, we further address the data reliable delivery through a simple but effective way. From the security analysis, our protocol ARSC is more secure than other schemes such as ANODR, SDAR, AnonDSR, CAR and MASK. Also, our scheme ARSC has better performance than any other onion-structured anonymous routing protocols such as AnonDSR.

## ACKNOWLEDGMENTS

This work is supported by National Natural Science Foundation of China under contract 60902008, and the Program of Changzhou Key Laboratory of Hi-tech under contract CM20103003, and the Key Lab of Information Network Security, Ministry of Public Security under contract C12602.

## REFERENCES

[1] Tehrani, A.H., Shahnasser, H., "Anonymous Communication in MANET's, Solutions and Challenges", IEEE International Conference on Wireless Information Technology and Systems (ICWITS), pp.1-4, San Francisco, CA, USA , August, 2010.

[2] J. Kong and X. Hong. "ANODR: Anonymous On Demand Routing with Untraceable Routes for Mobile Ad-hoc Networks", MobiHoc'03, Annapolis, USA, 2003.

[3] R. Dingledine, N. Mathewson, and P. Syverson. "Tor: Tsecond-generation onion router". Proceedings of the 13 USENIX Security Symposium, Aug 2004.

[4] D. Goldschlag, M. Reed, and P. Syverson. "Onion routing fanonymous and private internet connections". Communictions of the ACM, 1999.

[5] A. Boukerche, K. EI-Khatib, L. Xu, and L. Korba. "SDAR: A Secure Distributed Anonymous Routing Protocol for Wireless and Mobile Ad Hoc Network", The 29th Annual IEEE International Conference on Local Computer Networks, Tampa, Florida, USA, 2004 .

[6] R. Song, L.Korba, and G. Yee. "AnonDSR: Efficient Anonymous Dynamic Source Routing for Mobile Ad-Hoc Networks", The Third ACM Workshop on Security of Ad Hoc and Sensor Networks, Alexandria, VA, USA, 2005.

[7] R. Song, L.Korba. "A Robust Anonymous Ad Hoc On demand Routing", MILCOM, October 18, 2009.

[8] Reza Shokri, Nasser Yazdani, Ahmad Khonsari. "Chain-based Anonymous Routing for Wireless Ad_Hoc Networks", Consumer Communications and Networking Conference, Las Vegas, NV, USA, 2007.

[9] Jung Ha Paik, Bum Han Kim, and Dong Hoon Lee. "A3RP : Anonymous and Authenticated Ad Hoc Routing Protocol", Proceedings of International Conference on Information Security and Assurance, 2008.

[10] Y. Zhang, W. Lou, and W. Lou. Anonymous Communications in Mobile Ad Hoc Networks, The 24th IEEE Annual Conference on Computer Communications (INFOCOM'05), Miami, FL, USA, March 13-17, 2005.

[11] El Defrawy, K., Tsudik, G. "Privacy-Preserving Location-Based On-Demand Routing in MANETs". IEEE Journal on Selected Areas in Communications, Vol.29, No.10, pp.1926-1934, December 2011.

[12] Carlos T. Calafate, Javier Campos, Marga Nácher, Pietro Manzoni, Juan-Carlos Cano. "A-HIP: A Solution Offering Secure and Anonymous Communications in MANETs". Advance in Information and Computer Security, 2010, Volume 6434/2010, 217-231.

[13] Kashyap Balakrishnan, Jing Deng, Pramod K. Varshney. "TWOACK: Preventing Selfishness in Mobile Ad Hoc Networks", IEEE Wireless Communications and Networking Conference, pp.2137-2142, 2005.

[14] Panagiotis Papadimitratos, Zygmunt J. Haas. "Secure Data Communication in Mobile Ad Hoc Networks", IEEE Journal on Selected Areas in Communications, vol.24, no.2, pp.343-356, 2006.

[15] Li Zhao, José G. Delgado-Frias. "Multipath Routing Based Secure Data Transmission in Ad Hoc Networks", WiMob, pp.17-26, 2006.

[16] Heesook Choi, William Enck, Jaesheung Shin, and Patrick D. McDaniel. "ASR: anonymous and secure reporting of traffic forwarding activity in mobile ad hoc networks", Wireless Networks, Vol.15, 2009.

[17] W. Dai, Speed Comparison of Popular Crypto Algorithms, crypto++ Library 5.6.0 Benchmarks, 2009.

[18] Christoph Sommer, Isabel Dietrich and Falko Dressler, Simulation of Ad Hoc Routing Protocols using OMNeT++, Mobile Networks and Applications, vol.15, no.6, pp.786-801, 2011