Wireless Netw (2011) 17:843–859
DOI 10.1007/s11276-010-0319-2

AOS: an anonymous overlay system for mobile ad hoc networks

Rui Zhang · Yanchao Zhang · Yuguang Fang

Published online: 14 January 2011

Abstract Providing anonymous communications in mobile ad hoc networks (MANETs) is an effective countermeasure against malicious traffic analysis. This paper presents AOS, an Anonymous Overlay System for MANETs, which provides provably strong source and destination anonymity under a rather strong adversary model. AOS differs significantly from previous anonymous communication systems for MANETs mainly in three aspects. First, AOS is an overlay system independent of the underlying MANET protocol stack. Second, AOS resolves the conflict between anonymous communications and secure routing in MANETs and enables providing both at the same time. Last but not least, AOS can satisfy diverse anonymity requirements with different communication and computation overhead. AOS is the first system of its kind, and its efficacy and efficiency are confirmed by detailed qualitative and quantitative analysis.

Keywords Ad hoc networks · Anonymous communication · Onion routing · Overlay

R. Zhang · Y. Zhang
School of Electrical, Computer, and Energy Engineering, Ira A. Fulton Schools of Engineering, Arizona State University, Tempe, AZ 85287-5706, USA

Y. Fang (✉)
Department of Electrical and Computer Engineering, University of Florida, 435 Engineering Building, P.O. Box 116130, Gainesville, FL 32611, USA
e-mail: fang@ece.ufl.edu

Y. Fang
Changjiang Scholar Chair Professor, National Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China

1 Introduction

The inherent broadcast nature of mobile ad hoc networks (MANETs) facilitates adversarial eavesdropping on data transmissions. In particular, an attacker can surreptitiously intercept all the packets within the reception range of his radio transceiver without causing attention. If there are multiple attackers with powerful enough transceivers, they may collaboratively record all the packets sent anytime, anywhere in the network. A packet consists of a header and a payload part. Packet payloads can be encrypted to prevent attackers from knowing the information carried in intercepted packets, while packet headers have to be left in clear to enable multi-hop routing. Attackers thus can learn packet sources and destinations in packet headers. One may consider letting neighboring nodes establish pairwise link-layer keys whereby to encrypt packet sources and destinations. This countermeasure, however, fails in the presence of compromised nodes.

The ability of attackers to infer real packet sources and destinations enables *traffic analysis* [1] run to infer the network traffic pattern and its changes. A network traffic pattern consists of triplets ⟨source, destination, average rate⟩, each describing one flow [2]. In a tactical military MANET, an abnormal change of the network traffic pattern may indicate a forthcoming action, a chain of commands, or a state change of network alertness [3]. Its disclosure to attackers may thus lead to the failure of urgent military actions. In addition, in many cases packet sources are VIP nodes such as captains, while packet destinations are nodes commanded to carry out certain critical operations. If noticing that some nodes often act as packet source or destinations, attackers may consider these nodes important and launch pinpoint attacks on them.

Providing communication anonymity has long been recognized as an effective defense against traffic analysis

[4–6]. As discussed in [5], communication anonymity has two essential requirements, *source anonymity* and *destination anonymity*, which indicate the impossibility to identify the source and the destination of any given packet, respectively. Another often-mentioned anonymity requirement is *relationship anonymity*, which means that it is untraceable who communicates with whom. It is weaker than and can be derived from each of source and destination anonymity [7]. Traffic analysis can obviously be thwarted in an anonymous communication system, as attackers can no longer ascertain true sources and destinations of intercepted packets.

Tremendous efforts (e.g., [8–21]) have been made on anonymous communication in MANETs, but many challenges still remain to be tackled. First of all, previous solutions all employ a distinct, non-standard routing protocol specifically tailored for anonymous communication. This makes it impossible to harness the advance in non-anonymous MANET routing protocols that focus on optimizing routing efficiency. Second, as a consequence of providing communication anonymity at the network layer, none of previous solutions are compatible with the non-repudiation requirement of a secure MANET routing protocol, thus vulnerable to various routing attacks such as wormhole [22] and pushing [23] attacks. In particular, secure MANET routing protocols (e.g., [22–26]) require that each node be accountable for any routing packet sent from it, while existing anonymous communication solutions all require that mobile nodes avoid being identified while participating in routing operations. Last but not least, previous solutions fail to provide differentiated anonymity to different MANET nodes, which is nevertheless desired in many practical scenarios. For example, a commander node in a military MANET may always need higher anonymity protection than ordinary nodes when sending or receiving a message. Since stronger anonymity is often associated with larger communication and computation overhead, previous solutions that treat all the nodes equally must thus satisfy the highest anonymity requirement among all the nodes. This may obviously result in considerable wastage of network resources.

In this paper, we propose AOS, a novel *A*nonymous *O*verlay *S*ystem for MANETs to hide real packet sources and destinations among crowds of nodes. In particular, AOS organizes MANET nodes into pairwise-disjoint sets called *clique*s. Members of each clique exchange encrypted traffic of adjustable rate into which dummy and real data packets can be inserted. To ensure strong communication anonymity, the source, say S, no longer sends packets along the shortest path to the destination, say D. Instead, S randomly selects a tree of cliques, one of which contains D. Each packet from S will be sent along a random path within every clique of the chosen clique tree and can eventually reach the intended destination D. AOS provides provably strong source and destination anonymity against both global eavesdroppers (external attackers) and internal attackers at the cost of increased communication and computation overhead, which is fortunately tunable according to different levels of anonymity requirement.

As the first work of its kind, AOS distinguishes itself from previous anonymous communication solutions (e.g., [8–21]) for MANETs in the following aspects:

- *Universal applicability*: AOS is an anonymous network overlay atop the MANET substrate and has no special requirement for the underlying MANET protocol stack. This feature would apparently enhance its applicability.
- *Coexistence with secure routing*: AOS is not a network-layer solution as compared to previous work, so it can be built upon any MANET secure routing protocol to ensure both communication anonymity and routing security.
- *Differentiated anonymity provision*: AOS can satisfy diverse anonymity requirements with different communication and computation overhead.

The rest of this paper is organized as follows. Section 2 reviews the related work and Section 3 gives the network and adversary models as well as our design objectives. Section 4 presents the AOS design, followed by detailed performance evaluation in Section 5. This paper is finally concluded in Section 6.

2 Related work

Due to the space limitation, in this section we only discuss the prior research closely related to our work.

The seminar work on anonymous wired communication was done by Chaum who proposed the concept of Mix-Nets [4] for anonymous email. In a Mix-Net, each message is sent through a series of pre-deployed stations, called *mixes*, to the final destination. The Mix-Net design has inspired many anonymous communication systems, e.g., Crowds [6] and Onion Routing [27]. In Crowds, each web request travels a random path whose length follows a given geometric distribution to the end server. In Onion Routing [27], a special data structure called *onion* is formed by the source with multiple layers of encryption. Each intermediate router, only knowing its predecessor and successor, peels off one layer of encryption, and finally the receiver obtains the packet in plaintext. Our AOS can be considered as a unique combination of Mix-Net [4], Crowds [6], and Onion Routing [27, 28] specifically tailored for MANETs. In particular, Crowds and Onion Routing are used for intra-clique and inter-clique communications, respectively. This design leads to another nice feature. Since each message

may traverse a clique through different number of nodes, each clique in fact serves as a virtual *mix*, through which messages are mixed. In addition, by adopting the technique proposed in [28], any intermediate node cannot determine its relatively position in the whole path by observing the message length, format, or content.

Providing anonymous communications in MANETs has also been investigated extensively, see [8–21] for example. As discussed in Sect. 1, all these schemes suffer from the lack of universal applicability, the conflict with secure routing, and the lack of differentiated anonymity provision. In contrast, our AOS overcomes these drawbacks at the cost of increased communication overhead because it is an application-layer solution and not intended to optimize the routing efficiency.

3 Models and design goals

3.1 Network model

We consider a MANET of N nodes controlled by a single authority. MANETs of this type have long been recognized as a major application category of wireless ad hoc networking techniques. Typical examples are those deployed in military and counter-terrorist operations. Since mobile nodes have common interests in such a network, they are not selfish [29] and readily forward packets to and from others.

We assume that each node has limited transmission and reception capabilities. Two nodes out of transmission range of each other can communicate via a sequence of intermediate nodes in a multihop fashion. AOS is an anonymous network overlay atop the MANET substrate and has no requirement for the underlying MANET protocol stack. We, however, do assume that a valid unicast route can be established between any two nodes when needed, and reestablished when it breaks due to node mobility.

3.2 Adversary model

We focus on dealing with an adversary whose sole goal is to find out real sources and/or destinations of intercepted packets whereby to infer the network traffic pattern and its changes. It is beyond the scope of this paper to address other important security issues in MANETs such as secure routing, key management, intrusion detection, and DoS mitigation.

The adversary consists of a *global external attacker*, which does not belong to the target MANET but can passively eavesdrop on all the radio transmissions, and some *local internal attackers*, which are a small fraction of MANET nodes compromised and fully controlled by the adversary. Moreover, the adversary is assumed to be computationally bounded and cannot break any cryptographic primitive on which we base in our design.

3.3 Design objectives

AOS is designed to provide strong source and destination anonymity as well as relationship anonymity under the above rather strong adversary model. In particular, AOS is intended to satisfy the following anonymity requirements [7]:

- *Source anonymity*: Given a packet, it is impossible to identify its real source. Alternatively, given a node, it is impossible to tell what packets it initiated.
- *Destination anonymity*: Given a packet, it is impossible to ascertain its real destination. Alternatively, given a node, it is infeasible to tell what packets are destined to it.
- *Relationship anonymity*: It is impossible to tell whether and when any two nodes communicate.

Relationship anonymity is weaker than the previous two because it can be guaranteed as long as either or both of source and destination anonymity are realized. Therefore, we will focus on source and destination anonymity hereafter.

3.4 Notations

Table 1 summarizes the notations to be used throughout the paper.

Table 1 Some probabilistic parameters

Notation	Meaning
S, D	Source and destination
$\mathcal{C}_i$	The ith clique
$C_{i,j}$	The network ID of the jth node in clique $\mathcal{C}_i$
$\mathbb{G}_1$	An additive group of order q
$\mathbb{G}_2$	An multiplicative group of order q
$\mathcal{K}_A$	The private key of node A
K_{AB}	The share key between nodes A and B
α	The total number of onion layers
β	The total number of onion cliques
β_i	The number of onion cliques in the ith layer
$\mathcal{A}_{i,j}$	The jth onion clique in the ith layer
$O_{i,j}$	The onion node of $\mathcal{A}_{i,j}$
$P_{i,j}$	The proxy node of $\mathcal{A}_{i,j}$
$\mathcal{P}_{S,i}$	The pseudonyme of S for the ith layer
K_i	The shared key between S and node $O_{i,1}$
K_D	The shared key between S node D
$\mathcal{Q}_i$	The ith path object
$\mathcal{O}_i$	The onion before entering the ith layer

4 AOS: an anonymous overlay system

In this section, we first outline the basic idea of AOS and then present its design in detail.

4.1 An overview of AOS

In AOS, MANET nodes are divided into pairwise-disjoint sets called *cliques*, and each node knows which clique any other node belongs to. We take source S and destination D as an example to outline the basic AOS operations. For simplicity, we temporarily assume that S and D are in different cliques.

To anonymously send messages to D, node S randomly selects a few distinct cliques called *onion cliques* hereafter, one of which contains D. Node S then picks one node in every onion clique which is called an *onion node* through which every message will be routed. AOS adopts the idea of mix-nets [4]. In particular, S wraps the message for D in several layers of encryption. The wrapped message, often referred to as an onion [27], is then routed through onion nodes each of which peels off a layer of encryption and then forwards the onion to the next onion node. By forming the onion properly, we can ensure that every onion node knows nothing more than its next onion node: it knows neither other onion nodes nor how many onion nodes separate it from S or D. Even the onion node prior to D does not know that D is actually the destination. Source S and destination D are thus hidden from all the onion nodes.

AOS integrates a number of techniques to defend against both external and internal attackers. For example, to mask its transmission behavior, S does not directly send the onion to the first onion node; instead, the onion will be first sent to a randomly chosen clique peer. Then, each node, on receiving a onion from its clique peer, will send the onion to the first onion clique with some probability, or randomly choose another clique peer to which the onion is sent. Finally, the onion will be sent to a randomly chosen *proxy node* in the first onion clique other than the first onion node, which in turn forwards the onion to the first onion node. We will show that this simple measure helps hiding the first onion node from external attackers. The onion will be forwarded in subsequent onion cliques in a similar way. To further prevent attackers from tracing the onion, AOS requires the members of every clique to exchange encrypted cover traffic of adjustable rate into which onions (real packets) and dummy packets can be inserted. Furthermore, S chooses some fake onion cliques for each true onion clique, from each of which to select a fake onion node for the corresponding true onion node. Destination D can then be hidden among these fake and real onion nodes. All these measures are intended to make it very difficult for the adversary to determine packet sources and destinations at the cost of tunable communication overhead.

An example is given in Fig. 1, where each real onion clique $\mathcal{A}_{i,1}(1 \le i \le 3)$ together with some fake ones form one onion layer. We denote by $O_{i,j}$ the onion node in each $\mathcal{A}_{i,j}$, and only $O_{i,1}(1 \le i \le 3)$ are real ones. Destination D is in clique $\mathcal{A}_{2,2}$ and is actually $O_{2,2}$. In addition, nodes $P_{i,j}$ in each clique are proxy nodes. As we can see, source S forms an onion which is routed through four random nodes and

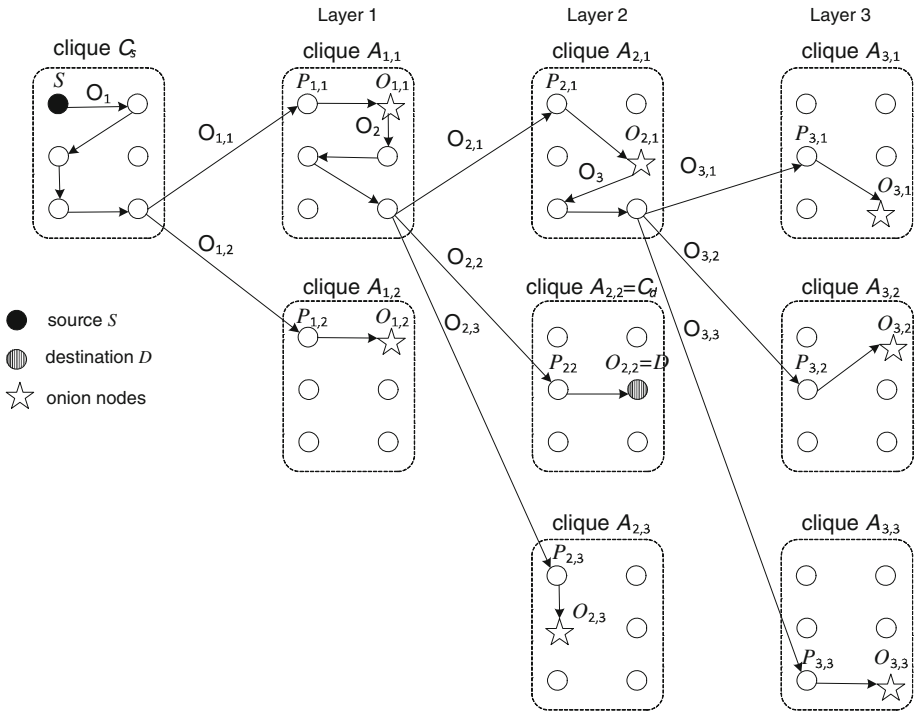


Fig. 1 An exemplary illustration of AOS

then forwarded to each $P_{1,j}$ which in turn sends the onion to $O_{1,j}$. Only the real onion node $O_{1,1}$ peels off one layer of encryption, and then the modified onion passes three random nodes to reach the next onion layer. This process continues until $O_{3,1}$ terminates the onion forwarding. D apparently can receive the message while hiding in all the onion nodes. AOS can also ensure that each onion node cannot determine which onion layer it resides at, i.e., its distance from the source or destination clique. This onion-forwarding process will be more clear when we come back to it in Sect. 4.7.

In the remainder of this section, we will illustrate the design details of AOS, including clique formation, intra-clique cover traffic, key distribution and agreement, and onion construction and forwarding.

4.2 Clique formation

We consider a single-authority MANET with N nodes. Before the network deployment, the network owner divides N nodes into M pairwise-disjoint cliques denoted by $\{\mathcal{C}_0, \ldots, \mathcal{C}_{M-1}\}$, that is,

$$\begin{cases} \mathcal{C}_i \cap \mathcal{C}_j = \phi, & \forall i,j \in \{0,\ldots,M-1\}, i \neq j, \\ \sum_{i=0}^{M-1} |\mathcal{C}_i| = N. \end{cases}$$

Note that the cliques need not have the same number of nodes. In addition, cliques have only virtual meanings: nodes of the same clique may be physically apart during the network operations. Nodes in each clique $\mathcal{C}_i$ are indexed from 0 to $|\mathcal{C}_i| - 1$. Let $C_{i,j}$ denote the network ID (or address) of the jth node in clique $\mathcal{C}_i$, where $0 \le j \le |\mathcal{C}_i| - 1$. The network owner preloads each node with the information regarding the affiliated clique 2, the index in that clique, and the network ID of every other node.

4.3 Key distribution and agreement

Like other security schemes, AOS requires nodes to have appropriate cryptographic keys. In this paper, we assume a key distribution scheme (e.g., [30]) based on Identity-Based Cryptography (IBC) [31]. AOS, however, can also rest on other suitable key distribution schemes.

Since AOS targets a single-authority MANET, it is reasonable to assume that a trusted authority (TA) will bootstrap the network, which itself is not part of the resulting network. The TA chooses a large prime q, a master secret $\kappa \in \mathbb{Z}_q^*$, an additive group $\mathbb{G}_1$ of order q, a multiplicative group $\mathbb{G}_2$ of order q, a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, and a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{G}_1^*$ which maps arbitrary binary strings into nonzero points in $\mathbb{G}_1$. Modified Weil [31] and Tate [32] pairings are examples of the bilinear map $\hat{e}$, and we refer the readers to [31, 32] for the detailed properties of $\hat{e}$. The TA equips each node with $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, \mathcal{H} \rangle$ while keeping κ confidential to itself.

Each node A has a unique public/private key pair, where the public key is its unique network ID, i.e., A, and the private key $\mathcal{K}_A = \kappa \mathcal{H}(A)$ is obtained from the TA. Note that the master secret κ cannot be recovered from any public/private key pair like $A/\mathcal{K}_A$ [30–32], so it is always known only to the TA.

In AOS, any two nodes, say A and B, can establish a shared key without interacting with each other. In particular, A and B independently compute

$$K_{AB} = \hat{e}(\mathcal{K}_A, \mathcal{H}(B)) \quad \text{and} \quad K_{BA} = \hat{e}(\mathcal{K}_B, \mathcal{H}(A)). \tag{1}$$

It can be shown that $K_{AB} = K_{BA}$ are equal due to the bilinear and symmetric properties of $\hat{e}$ [30–32].

Each node A can also generate arbitrarily many pseudonyms and the corresponding private keys. For example, A can select a random integer $r_A \in \mathbb{Z}_q^*$ whereby to compute a pseudonym $r_A \mathcal{H}(A) \in \mathbb{G}_1$ and the corresponding private key $r_A \mathcal{K}_A = r_A \kappa \mathcal{H}(A)$. Since $\mathbb{G}_1$ is a cyclic group of order q, multiplying $\mathcal{H}(A)$ by r_A perfectly blinds $\mathcal{H}(A)$ and A [33]. In other words, given only the pseudonym $r_A \mathcal{H}(A)$, it is infeasible to link it to node A. The use of such pseudonyms can be seen soon.

4.4 Intra-clique and inter-clique traffic

Since each MANET node serves as a router to relay packets to and from other nodes, it seems that MANET provides natural source anonymity because attackers are unable to differentiate whether a target node has initiated or just forwarded a given packet. Unfortunately, this argument holds only when the incoming traffic rate of the target node is not smaller than its outgoing traffic rate. Otherwise, attackers can determine that the node has initiated some traffic. This may be dangerous even if attackers cannot differentiate what packets were initiated by that node. For example, a node determined to initiate more packets than others is highly possible to be a VIP node. AOS thus must prevent this from occurring.

AOS uses *cover traffic* to cloak packet sources. In particular, any two nodes in clique $\mathcal{C}_i$, for all $0 \le i \le M-1$, exchange cover packets at a rate of λ packets/second, where λ is a public system parameter. Each cover packet might be a real data packet or a dummy packet (inserted to maintain the constant traffic rate). To prevent attackers from distinguishing data and dummy cover packets, we require that cover packets be of equal length and that a cover packet between any two nodes be encrypted using their shared key established as in Eq. (1).

Consider as an example nodes A and B both in clique $\mathcal{C}_i$ which exchange cover packets as follows.

$$A \leftrightarrow B: \{I|t|\text{DATA}\}_{K_{AB}}, h_{K_{AB}}(\text{prior-data}),$$

where $\{\cdot\}_*$ denotes a length-preserving symmetric-key encryption using the key on the subscript; K_{AB} is the shared key of A and B using Eq. (1); I is a one-bit indicator of whether this packet is a dummy ($I = 0$) or data ($I = 1$) packet; t is the timestamp for guaranteeing message freshness; DATA of constant length contains either useful or dummy data, $h_{K_{AB}}(\cdot)$ is a keyed hash function to ensure message authenticity. Upon receiving the cover packet, node A (or B) verifies the message authentication code. If succeed, it then uses K_{AB} to decrypt message. If $I = 0$ (a dummy packet), A (or B) drops the packet; otherwise, A (or B) processes DATA using the method in Sect. 4.6.

Global external attackers cannot determine whether node A initiated a data packet, as they cannot differentiate data cover packets from dummy cover packets. AOS also ensures that even node B cannot determine whether DATA was initiated or just forwarded by node A and thus protects A from internal attackers if any.

In addition, λ determines the maximum throughput of every node in clique $\mathcal{C}_i$. In particular, the maximum traffic rate generated by every node is $\lambda(|\mathcal{C}_i|)$ packets/second, which occurs when the cover packets to all its clique peers are all data packets initiated by this node. The larger λ, the higher the maximum throughput, and the larger the communication overhead associated with inserting dummy cover packets to maintain the constant cover-traffic rate λ between any two nodes in $\mathcal{C}_i$.

MANET nodes can dynamically adjust λ to balance between the maximum throughput and the communication overhead. For example, λ can be set small most of the time. When some node, say A, desires higher throughput for a certain period, it can specify its demand in a message anonymously delivered to a random destination, say B, using the method in Sect. 4.5. B then floods the request throughout the network, upon receipt of which each node increases λ to the required value and changes it back after the specified period. This measure prevents attackers from detecting A's intention. The downside is that attackers may infer some information from the changes in the cover-traffic rate.

In contrast to intra-clique traffic, inter-clique traffic consists of only real data packets which are sent on demand. Inter-clique packets have the same format as intra-clique cover packets with $I = 1$. Therefore, intra-clique and inter-clique packets are of the same fixed length, which can prevent attackers from inferring any useful information from packet-length changes [1, 28].

4.5 Onion construction

Now we discuss the construction of onions. Our onion construction differs from the original onion routing [27, 28] in that each onion layer consists of a variable number of onion nodes instead of only one. Assume that source $S \in \mathcal{C}_s$ intends to send a message msg to destination $D \in \mathcal{C}_d$, where s and d might be equal or not. To construct an onion, S first does the following.

1. Select an integer $\beta \in [\alpha, M]$ as the total number of onion cliques distributed across the α onion layers, where α is a public system parameter. How to choose α and β will be discussed in Sect. 5.1.
2. Generate α random numbers $\beta_1, \ldots, \beta_\alpha$ such that $\beta_i \geq 1$ and $\sum_{i=1}^{\alpha} \beta_i = \beta$, where β_i is the number of onion cliques at onion layer i.
3. Choose the onion cliques for each onion layer. In particular, randomly select β_i cliques with consecutive indexes as the onion cliques of each layer i, denoted by $\{\mathcal{A}_{i,j}\}_{j=1}^{\beta_i}$. Let $\mathcal{A}_{i,j} = \mathcal{C}_{x_{i,j}} \in \{\mathcal{C}_z\}_{z=1}^{M-1}$. For example, if clique $\mathcal{C}_z$ is chosen uniformly at random from $\{\mathcal{C}_z\}_{z=0}^{M-1}$ as $\mathcal{A}_{i,1}$, then we have $\mathcal{A}_{i,j} = \mathcal{C}_{(z+j-1) \mod M}, 2 \leq j \leq \beta_i$. Only cliques $\{\mathcal{A}_{i,1}\}_{i=1}^{\alpha}$ are real onion cliques, while others are fake. In addition, a clique can appear at multiple onion layers. The only requirement is that destination clique $\mathcal{C}_d$ be chosen at least once as either a real or fake onion clique.
4. For each onion clique $\mathcal{A}_{i,j}, 1 \leq i \leq \alpha, 1 \leq j \leq \beta_i$, choose one onion node, denoted by $O_{i,j}$. Let $O_{i,j} = C_{x_{i,j},y_{i,j}}$ (cf. Sect. 4.2), where $x_{i,j}$ has been chosen in Step 3, so only node index $y_{i,j}$ need be determined. Here one trick is to select onion nodes with correlated indexes such that the β_i onion nodes at each layer i can be represented by a triplet of constant length, which further helps maintain a constant onion size (as will be shown later). Specifically, since destination clique $\mathcal{C}_d$ may be chosen multiple times, assume that it first appears at layer l and is actually onion clique $\mathcal{A}_{l,e}$ (i.e., $x_{l,e} = d$). Then destination D should be chosen as the onion node $O_{l,e}$ of $\mathcal{A}_{l,e}$. Assume that the index of $O_{l,e}$ in clique $\mathcal{A}_{l,e}$ is $y_{l,e}$, where $0 \leq y_{l,e} \leq |\mathcal{A}_{l,e}| - 1$. The onion nodes are selected as follows.
   - For each layer $i \neq l$, i.e., the one does not contain $\mathcal{C}_d$, first randomly select $y_{i,1} \in [0, |\mathcal{A}_{i,1}| - 1]$ and an integer v_i. Then compute
     $$y_{i,j} = v_i y_{i,1} \mod |\mathcal{A}_{i,j}|, \quad 2 \leq j \leq \beta_i .$$
   - For layer l, there are two cases:
     - If $e = 1$, i.e., $\mathcal{C}_d$ is the first onion clique in layer l, randomly select an integer v_l. Then compute
       $$y_{l,j} = v_l y_{l,1} \mod |\mathcal{A}_{l,j}|, \quad 2 \leq j \leq \beta_l .$$
     - If $e \neq 1$, first randomly select $y_{l,1} \in [0, |\mathcal{A}_{i,1}| - 1]$ and an integer v_l such that $y_{l,e} = v_l y_{l,1} \mod |\mathcal{A}_{l,e}|$. Then compute

$$y_{l,j} = v_i y_{l,1} \mod |\mathcal{A}_{l,j}|, \quad 2 \le j \le \beta_l .$$

By doing so, given a triplet $\langle O_{i,1}, v_i, \beta_i \rangle$, the node indexes of all the onion nodes in layer i can be derived as $y_{i,j} = v_i y_{i,1} \mod |\mathcal{A}_{i,j}|, 2 \le j \le \beta_i$. In other words, the β_i onion nodes at each layer i can be represented with the triplet $\langle O_{i,1}, v_i, \beta_i \rangle$, which is of constant length.

5. Select a unique random integer $r_i \in \mathbb{Z}_q^*$ for each onion layer i whereby to compute a pseudonym $\mathcal{P}_{S,i} = r_i \mathcal{H}(S) \in \mathbb{G}_1$ and the corresponding private key $r_i \mathcal{K}_S = r_i \kappa \mathcal{H}(S) = s\mathcal{P}_{S,i}$ (cf. Sect. 4.3). $\mathcal{P}_{S,i}$ is used to hide source S from onion nodes at layer i, as will be shown soon.
6. Calculate a shared key with each $O_{i,1}, i \in [1, \alpha]$, as $K_i = \hat{e}(r_i \mathcal{K}_S, \mathcal{H}(O_{i,1})) = \hat{e}(r_1 \kappa \mathcal{H}(S), \mathcal{H}(O_{i,1}))$, which will be used to add (by S) or strip off (by $O_{i,1}$) a layer of encryption.

To prevent each onion node from knowing its distance from source S or destination D, i.e., which onion layer it is at, we use the following approach. Assume that destination D is chosen as the onion node $O_{l,e}, l \in [1, \alpha]$.[1] Source S calculates a shared key with D as $K_D = \hat{e}(r_d \mathcal{K}_S, \mathcal{H}(D))$ and then computes

$$\mathcal{M} = \begin{cases} \{msg\}_{K_D} & l = 1, \\ \{\{msg\}_{K_D}\}_{K_1} & l = 2, \\ \{\{\ldots\{\{msg\}_{K_D}\}_{K_{l-1}}\ldots\}_{K_2}\}_{K_1} & 2 < l \le \alpha. \end{cases} \quad (2)$$

Source S proceeds to derive

$$\mathcal{Q}_i = \begin{cases} TAG_1 & i = 1, \\ \{TAG_2\}_{K_1} & i = 2, \\ \{\ldots\{\{TAG_i\}_{K_{i-1}}\}_{K_{i-2}}\ldots\}_{K_1} & 3 \le i \le \alpha, \end{cases} \quad (3)$$

where $TAG_i = Flag|\mathcal{P}_{S,i}|O_{i,1}|v_i|\beta_i$. Here, $Flag$ is a predetermined public string that indicates the legitimacy of the TAG_i, which will be more clear later. In addition, since $\{\cdot\}_*$ is a length-preserving symmetric-key cipher, we have $|\mathcal{Q}_1| = |\mathcal{Q}_2| = \cdots = |\mathcal{Q}_\alpha|$. Each $\mathcal{Q}_i$ is called a *path object* that carries information about the onion path.

Finally, S forms the onion as

$$\mathcal{O}_1 = \langle \mathcal{M}, \mathcal{Q}_\alpha, \mathcal{Q}_{\alpha-1}, \ldots, \mathcal{Q}_2, \mathcal{Q}_1 \rangle . \quad (4)$$

If S has another message for the same destination D, it can form a new onion by just changing the message part $\mathcal{M}$. In other words, the same set of onion nodes can be used in multiple messages between S and D. S, however, still needs to update the set of onion nodes periodically if it has a large number of messages for D to prevent the predecessor attack [34].

[1] If D is chosen as an onion node at multiple layers, we select l as the smallest layer.

4.6 Onion forwarding and processing

Onion $\mathcal{O}_1$ takes a random path in source clique $\mathcal{C}_s$ before entering onion layer 1. Specifically, source S picks a random node from $\mathcal{C}_s$ (possibly itself) and sends $\mathcal{O}_1$ to it in a cover packet (cf. Sect. 4.4). When the chosen node receives $\mathcal{O}_1$, it forwards $\mathcal{O}_1$ with probability $\eta \in [0, 1)$ to a random node in $\mathcal{C}_s$ (possibly itself) and with probability $1 - \eta$ directly to onion layer 1. η, called the *mixing probability*, is a public system parameter whose choice will be discussed in Sect. 5. Borrowed from Crowds [6], this random-forwarding technique helps preventing the adversary from identifying source S, whose optimality has recently been proved in [35]. In particular, every onion forwarder (except S) cannot tell whether the clique peer sending $\mathcal{O}_1$ is the onion source or just another random onion forwarder. We will show in Sect. 5 that this measure can provide strong source anonymity against both external and internal attackers.

Once some node in $\mathcal{C}_s$, say G, decides to forward $\mathcal{O}_1$ to onion layer 1, it first deduces the first-layer onion nodes $\mathcal{L}_1 = \{O_{1,1}, \ldots, O_{1,\beta_1}\}$ from $\langle O_{1,1}, v_1, \beta_1 \rangle$ in $\mathcal{Q}_1 = Flag|\mathcal{P}_{S,1}|O_{1,1}|v_1|\beta_1$. If G directly sends $\mathcal{O}_1$ to them, the adversary might easily ascertain $\mathcal{L}_1$ by passive eavesdropping. To prevent this, G picks a random node $P_{1,j}$ (called a *proxy node*) in each $\mathcal{A}_{1,j}, j \in [1, \beta_1]$, to which the following onion modified from $\mathcal{O}_1$ is sent.

$$G \to P_{1,j}: \ \mathcal{O}_{1,j} = \langle \mathcal{M}, \mathcal{Q}_\alpha, \ldots, \mathcal{Q}_2, Flag|\mathcal{P}_{S,1}|O_{1,j}|0|0 \rangle .$$

Note that v_i and β_i are replaced with zeros of equal length to hide them from $P_{1,j}$ (possibly compromised) and also maintain a constant onion length ($|\mathcal{O}_1| = |\mathcal{O}_{1,j}|$). $\mathcal{O}_{1,j}$ should be sent in an encrypted inter-clique packet (cf. Sect. 4.4) to $P_{1,j}$ which then directly forwards it in a cover packet to $O_{1,j}$ after checking $O_{1,j}$. Since external attackers cannot differentiate this onion cover packet from others between $P_{1,j}$ and $O_{1,j}$, they cannot immediately decide that $O_{1,j}$ is the onion node in clique $\mathcal{A}_{1,j}$. It is also possible that $P_{1,j}$ itself happens to be $O_{1,j}$, in which case $\mathcal{O}_{1,j}$ is directly processed as follows. The adversary cannot notice this either. If $P_{1,j}$ is a compromised node, then the adversary knows $O_{1,j}$ as the onion node, but he still cannot determine whether $O_{1,j}$ is the onion destination or identify other onion nodes at the same layer without knowing v_1 if other onion cliques have different sizes.

On receiving $\mathcal{O}_{1,j}$, each $O_{1,j}$ assumes that $\mathcal{O}_{1,j}$ was initiated by node $\mathcal{P}_{S,1}$ with which to calculate a shared key as $K_{1,j} = \hat{e}(\mathcal{K}_{O_{1,j}}, \mathcal{P}_{S,1})$. There are four cases:

– If $O_{1,j}$ is the real onion node, i.e., $j = 1$, then $K_{1,j} = \hat{e}(\mathcal{K}_{O_{1,1}}, \mathcal{P}_{S,1}) = \hat{e}(s\mathcal{H}(O_{1,1}), r_1 \mathcal{H}(S))$ which can be easily shown to be exactly K_1 [30–32].

- If $O_{1,j}$ is destination D, then $K_{1,j} = \hat{e}(\mathcal{K}_D, \mathcal{P}_{S,1}) = (s\mathcal{H}(D), r_1\mathcal{H}(S))$ which can be easily shown to be exactly K_D [30–32].
- If the previous two conditions are both meet, then we have $K_{1,j} = K_1 = K_D$.
- Otherwise, $K_{1,j}$ is equal to neither K_1 nor K_D.

Each $O_{1,j}$ then attempts using $K_{1,j}$ to decrypt $\mathcal{M}$ in $\mathcal{O}_{1,j}$, i.e., removing a layer of encryption. If the decryption result follows a predefined message format, then $O_{1,j}$ knows itself as the destination; otherwise, the decryption result can be ignored. Let $info^{x:y}$ denote the result of decrypting $info$ using shared keys $K_x, K_{x+1}, \ldots, K_{y-1}, K_y$ sequentially, which is performed by nodes $O_{x,1}, O_{x+1,1} \ldots, O_{y-1,1}, O_{y,1}$ sequentially. For example, $\mathcal{M}^{1:1}$ and $\mathcal{M}^{1:2}$ are the results of decrypting $\mathcal{M}$ using the shared key K_1 by $O_{1,1}$, and using K_1 and K_2 by $O_{1,1}$ and $O_{2,1}$ sequentially, respectively. According to Eq. (2), we have

$$\mathcal{M}^{1:1} = \begin{cases} \text{random string} & l = 1, O_{1,1} \neq D \\ msg & l = 1, O_{1,1} = D \\ \{msg\}_{K_D} & l = 2 \\ \{\ldots\{\{msg\}_{K_D}\}_{K_{l-1}}\ldots\}_{K_2} & 2 < l \leq \alpha\,. \end{cases} \tag{5}$$

In addition, each $O_{1,j}$ uses $K_{1,j}$ to decrypt $\mathcal{Q}_2 = \{Flag|\mathcal{P}_{S,2}|O_{2,1}|v_2|\beta_2\}_{K_1}$. Since $K_{1,1} = K_1$, only $O_{1,1}$ can do a successful decryption, which determines this after seeing $Flag$. In contrast, others know that they are fake onion nodes after not seeing $Flag$ in the decryption result and then stop processing $\mathcal{O}_{1,j}$. Node $O_{1,1}$ continues using K_1 to decrypt $\mathcal{Q}_3, \mathcal{Q}_4, \ldots, \mathcal{Q}_\alpha$, and obtain $\mathcal{Q}_3^{1:1}, \mathcal{Q}_4^{1:1}, \ldots, \mathcal{Q}_\alpha^{1:1}$. According to Eq. (3), we have $\mathcal{Q}_a^{1:1} = \{\ldots\{\{TAG_a\}_{K_{a-1}}\}_{K_{a-2}}\ldots\}_{K_2}, \forall a \in [3, \alpha]$. Finally, $O_{1,1}$ forms a new onion

$$\mathcal{O}_2 = \langle \mathcal{M}^{1:1}, R_1, \mathcal{Q}_\alpha^{1:1}, \ldots, \mathcal{Q}_3^{1:1}, \mathcal{Q}_2^{1:1} \rangle\,, \tag{6}$$

where $\mathcal{Q}_2^{1:1} = Flag|\mathcal{P}_{S,2}|O_{2,1}|v_2|\beta_2$, and R_1 is a random string of length $|\mathcal{Q}_1|$ inserted to compensate for the removal of $\mathcal{Q}_1$ so that $\mathcal{O}_2$ and $\mathcal{O}_1$ are of the same length and format. R_1 will be treated as a path object by subsequent onion nodes. Similar to $\mathcal{O}_1$, onion $\mathcal{O}_2$ takes a random path (starting from $O_{1,1}$) in clique $\mathcal{A}_{1,1}$ before reaching onion layer 2.

In general, each real onion node $O_{i,1}, i \in [1, \alpha - 1]$, generates onion $\mathcal{O}_{i+1}$ in which a random string R_i is inserted after the message part to maintain a constant onion length. R_is are indistinguishable from and will be processed as real path objects by subsequent onion nodes. In this way, each onion node (either real or fake) cannot determine at which onion layer it is. Our onion construction method is an adaptation of the one in [28] which we refer to for a formal security proof.

The onion forwarding is terminated at the last real onion node $O_{\alpha,1}$ which receives the following onion:

$$\mathcal{O}_{\alpha,1} = \langle \mathcal{M}^{1:(\alpha-1)}, R_{\alpha-1}, \ldots, R_1^{2:(\alpha-1)}, \mathcal{Q}_\alpha'^{1:(\alpha-1)} \rangle\,.$$

Here, $\mathcal{Q}_\alpha'^{1:(\alpha-1)} = Flag|\mathcal{P}_{S,\alpha}|O_{\alpha,1}|0|0$. Then $O_{\alpha,1}$ processes $\mathcal{O}_{\alpha,1}$ similarly as before. After using the shared key K_α to decrypt the rightmost "path object" $R_1^{2:(\alpha-1)}$, it does not find $Flag$ there and thus considers itself a "fake" onion node. Each fake onion node $O_{\alpha,j}, j \in [2, \beta_\alpha]$ also processes the received onion $\mathcal{O}_{\alpha,j}$ as before and makes the same decision. Note that all these nodes $O_{\alpha,j}, j \in [1, \beta_\alpha]$ cannot determine that they are at the last onion layer unless they collaborate to know that none of them further forwards the onion.

Now let us consider the processing of the message part M. The case of $l = 1$ has been considered in Eq. (5), so we focus on the more general case of $2 \leq l \leq \alpha$. From Eq. (2), we can derive that $\mathcal{M}^{1:(l-1)} = \{msg\}_{K_D}$. Same as before, each node $O_{l,j}, j \in [1, \beta_l]$ will attempt using a shared key to decrypt $\mathcal{M}^{1:(l-1)}$, and only destination D can correctly recover msg which has a predefined message format. If $l < \alpha$, each $\mathcal{M}^{1:(i-1)} (i \in (l, \alpha])$ will be determined by each onion node $O_{i,j} (j \in [1, \beta_i])$ as useless information after the decryption and thus simply ignored.

Since the same set of onion nodes is used in transmitting multiple messages from S to D, we can utilize this to reduce the computation overhead. In particular, each $O_{i,j}$ knows whether it is the destination or a real or fake onion node after processing the first message and can buffer the corresponding source pseudonym $\mathcal{P}_{S,i}$. If $\mathcal{P}_{S,i}$ appears in a later onion, only the destination and the real onion nodes process the onion using established shared keys, while other fake onion nodes simply ignore it.

4.7 An example

To shed more light on AOS, we continue the example in Fig. 1, where $\alpha = 3, \beta_1 = 2, \beta_2 = 3, \beta_3 = 3$, and destination D is actually the fake onion node $O_{2,2}$ in clique $\mathcal{A}_{2,2}$. For simplicity, all the cliques have the same size 6.

Source S generates $\mathcal{O}_1 = \langle \mathcal{M}, \mathcal{Q}_3, \mathcal{Q}_2, \mathcal{Q}_1 \rangle$, where

$$\begin{cases} \mathcal{M} = \{\{msg\}_{K_D}\}_{K_1} \\ \mathcal{Q}_3 = \{\{Flag|\mathcal{P}_{S,3}|O_{3,1}|v_3|3\}_{K_2}\}_{K_1} \\ \mathcal{Q}_2 = \{Flag|\mathcal{P}_{S,2}|O_{2,1}|v_2|3\}_{K_1} \\ \mathcal{Q}_1 = Flag|\mathcal{P}_{S,1}|O_{1,1}|v_1|2 \end{cases}$$

Here $v_i, i \in [1, 3]$ can be any integer due to the modulation. $\mathcal{O}_1$ passes through four nodes before leaving clique $\mathcal{C}_s$. Node $O_{1,j}$ ($j = 1, 2$) receives from its proxy node $P_{1,j}$ an onion $\mathcal{O}_{1,j} = \langle \mathcal{M}, \mathcal{Q}_3, \mathcal{Q}_2, Flag|\mathcal{P}_{S,1}|O_{1,j}|0|0 \rangle$ and computes a shared key $K_{1,j}$ based on the source pseudonym $\mathcal{P}_{S,1}$. It is

obvious that only $K_{1,1} = K_1$ with which $O_{1,1}$ generates $\mathcal{O}_2 = \langle \mathcal{M}^{1:1}, R_1, \mathcal{Q}_3^{1:1}, \mathcal{Q}_2^{1:1} \rangle$, where

$$\begin{cases} \mathcal{M}^{1:1} = \{msg\}_{K_D} \\ \mathcal{Q}_3^{1:1} = \{Flag|\mathcal{P}_{S,3}|O_{3,1}|v_3|3\}_{K_2} \\ \mathcal{Q}_2^{1:1} = Flag|\mathcal{P}_{S,2}|O_{2,1}|v_2|3\,. \end{cases}$$

$O_{1,2}$ knows itself as a fake onion node after decrypting $\mathcal{Q}_2$ with $K_{1,2}$ and not finding *Flag* there. Then it uses $K_{1,2}$ to decrypt $\mathcal{M}$. Since $K_{1,2} \neq K_D$, the decryption result is a random string which does not follow a predefined message format. So $O_{1,2}$ considers itself not the message destination.

$\mathcal{O}_2$ passes through three nodes before leaving clique $\mathcal{A}_{1,1}$. Node $O_{2,j}$ receives from its proxy node $P_{2,j}$ an onion $\mathcal{O}_{2,j} = \langle \mathcal{M}^{1:1}, R_1, \mathcal{Q}_3^{1:1}, Flag|\mathcal{P}_{S,2}|O_{2,j}|0|0 \rangle$ and then computes a shared key $K_{2,j}$ based on $\mathcal{P}_{S,2}$. This time we have $K_{2,1} = K_2$ and $K_{2,2} = K_D$. $O_{2,1}$ then uses $K_{2,1}$ to generate $\mathcal{O}_3 = \langle \mathcal{M}^{1:2}, R_2, R_1^{2:2}, \mathcal{Q}_3^{1:2} \rangle$, where $\mathcal{Q}_3^{1:2} = Flag|\mathcal{P}_{S,3}|O_{3,1}|v_3|3$. In contrast, $O_{2,2}$ and $O_{2,3}$ know that they are both fake onion nodes after decrypting $\mathcal{Q}_3^{1:1}$ with $K_{2,2}$ and $K_{2,3}$, respectively. Then they attempt decrypting $\mathcal{M}^{1:1}$ using $K_{2,2}$ and $K_{2,3}$, respectively. Node $O_{2,2}$ knows that it is the message destination, as the decryption result *msg* follows a predefined message format.

$\mathcal{O}_3$ passes through two nodes before leaving clique $\mathcal{A}_{2,1}$. Node $O_{3,j}$ receives from its proxy node $P_{3,j}$ an onion $\mathcal{O}_{3,j} = \langle \mathcal{M}^{1:2}, R_2, R_1^{2:2}, Flag|\mathcal{P}_{S,3}|O_{3,j}|0|0 \rangle$ and then computes a shared key $K_{3,j}$ based on $\mathcal{P}_{S,3}$. Each $O_{3,j}$ decrypts $R_1^{1:2}$ using $K_{3,j}$ and considers itself a fake onion node because *Flag* does not appear in the decryption result. Then it decrypts $\mathcal{M}^{1:2}$ with $K_{3,j}$ and knows that it is not the destination because the decryption result does not follow a predefined message format. Therefore, none of $O_{3,1}/O_{3,2}/O_{3,3}$ generates a new onion, so the onion forwarding stops. However, none of them can find out the termination of onion forwarding, i.e., that they are at the last onion layer, unless they collaborate.

Due to the use of pseudonyms, each $O_{i,j}$ thinks that it receives the onion $O_{i,j}$ from source $\mathcal{P}_{S,i}$. In addition, due to the insertion of R_1 by node $O_{1,1}$ and R_2 by $O_{2,1}$, the onion length keeps constant all the time. Therefore, each $O_{i,j}$ cannot determine which onion layer it resides in or how far it is from the source or destination: it could be at any layer $i \in [1, \alpha]$ with equal probability.

5 Performance analysis and evaluation

Although there has been a significant amount of work (e.g., [8–21]) on communication anonymity in MANETs, none of them has any of the three key good properties AOS has, i.e., differentiated QoA provision, universal applicability, and coexistence with secure routing (cf. Sect. 1). It is thus both unfair and less meaningful to conduct a performance comparison between AOS and existing work. Due to space limitations, we instead focus on thoroughly analyzing and evaluating the performance of AOS itself in this section. In particular, we first analyze its communication and computation overhead and then its security with regard to source and destination anonymity. Finally, we use numerical results to demonstrate the tradeoff between communication/computation overhead and source/destination anonymity.

5.1 Overhead analysis

5.1.1 Communication overhead

In AOS, a constant onion size L is used to prevent the adversary from deducing any useful information from onion-length changes. Each onion contains a fixed-length message part followed by α path objects of equal length. If the message is not long enough, it need be padded to maintain the fixed length. For simplicity, we choose $\mathcal{O}_1 = \langle \mathcal{M}, \mathcal{Q}_\alpha, \mathcal{Q}_{\alpha-1}, \ldots, \mathcal{Q}_2, \mathcal{Q}_1 \rangle$ to evaluate the *onion overhead* defined as the ratio of the total length of the non-message part to the onion length *len*. Since $|\mathcal{Q}_1| = |\mathcal{Q}_2| = \cdots = |\mathcal{Q}_\alpha|$, we just need to calculate $|\mathcal{Q}_1|$, where $\mathcal{Q}_1 = Flag|\mathcal{P}_{S,1}|O_{1,1}|v_1|\beta_1$.

We first discuss the length of $\mathcal{P}_{S,1}$, which is an element in group $\mathbb{G}_1$ (cf. Sect. 4.5) and more precisely a point on an elliptic curve over $\mathbb{F}_p$ [30–33]. If the prime p is of 171 bits and other pairing parameters are chosen properly, we can achieve a security level equivalent to that of 1024-bit RSA [33]. In addition, only the x-coordinate of $\mathcal{P}_{S,1}$ need be transmitted because the y-coordinate can be easily derived by solving the elliptic curve equation. This is known as the point compression technique. So we have $|\mathcal{P}_{S,1}| = 171$ bits.

Assume that each node ID is of l_{ID} bits, each v_i is of l_v bits, and each β_i is of ξ bits. We can derive the onion overhead as

$$\begin{aligned} \text{onionOverhead} &= \frac{\alpha|\mathcal{Q}_1|}{len} \\ &= \frac{\alpha(|\mathcal{P}_{S,1}| + |Flag| + |O_{1,1}| + |v_1| + |\beta_1|)}{len} \\ &= \frac{\alpha(171 + |Flag| + l_{ID} + l_v + \xi)}{len} \\ &= \frac{\alpha(171 + |Flag| + l_{ID} + l_v + \xi)}{|\mathcal{M}| + \alpha(171 + |Flag| + l_{ID} + l_v + \xi)} \\ &= \frac{1}{\rho + 1}, \end{aligned} \tag{7}$$

where $\rho = \frac{|\mathcal{M}|}{\alpha(171+|Flag|+l_{ID}+l_v+\xi)}$. In AOS, $|\mathcal{M}|$ is fixed and cannot be changed. Since $|Flag|, l_{ID}, l_v$ and ξ are also fixed,

the onion overhead is in direct ratio to α, the number of onion layers or path objects.

Now we examine totalTran, the total number of end-to-end packet transmissions incurred by each message from S to D, as AOS is an overlay system over the MANET substrate. For simplicity, we assume that no two consecutive onion forwarders are the same and that no onion node is chosen as a proxy node. Let L_i be the number of times for which onion $\mathcal{O}_i(\forall i \in [1, \alpha])$ is forwarded before reaching layer i. According to Sect. 4.6, L_i follows the geometric distribution $\mathbf{Pr}(L_i = k) = (1-\eta)\eta^{k-1}, \forall k \geq 1$, with mean $\bar{L}_i = \frac{1}{1-\eta}$. Each $\mathcal{O}_i$ also involves β_i inter-clique transmissions, each for one proxy node in layer i. Referring to the onion forwarding process in Sect. 4.6, we can easily calculate

$$\text{totalTran} = \sum_{i=1}^{\alpha} (\bar{L}_i + \beta_i) = \frac{\alpha}{1-\eta} + \beta\,. \tag{8}$$

Note that totalTran consists of $\frac{\alpha}{1-\eta} + \beta$ intra-clique transmissions and β inter-clique transmissions. The former are cloaked by intra-clique cover traffic at the rate of λ packets/second which can be dynamically adjusted as needed (cf. Sect. 4.4). Therefore, each message only incurs β additional inter-clique transmissions. In addition, the larger $\frac{\alpha}{1-\eta} + \beta$, the fewer concurrent sessions can be supported at a given value of λ, and vice versa.

In Sect. 5.3, we will use numerical results to show the tradeoff between communication overhead and source/destination anonymity.

5.1.2 Computation overhead

Now we discuss the computation overhead of AOS. The relatively most expensive cryptographic operation is the pairing $\hat{e}$ evaluation for shared-key establishment, which however can still be efficiently computed even on low-end devices. For example, the pairing evaluation on a Compaq iPaq 3660 PDA powered by a 206 MHz 32-bit StrongARM processor took just 355 ms [36]. Since AOS targets security-sensitive MANETs often with much more powerful mobile nodes, the pairing evaluation is expected to be completed within a few milliseconds [36].

The pairing function is also executed relatively rarely. In particular, each pair of nodes only need to execute $\hat{e}$ once on demand to establish a shared key whereby to encrypt and authenticate inter-clique or intra-clique traffic using efficient symmetric-key ciphers. Moreover, if source S intends to transmit a number of messages to destination D, each onion node (real or fake) only need compute $\hat{e}$ once to calculate a shared key when transmitting the first message. All subsequent onion processings are carried out based on the shared keys using efficient symmetric-key ciphers. Therefore, the computation overhead of AOS is quite acceptable in practice.

5.2 Security analysis

Packets in AOS no longer carry true source and destination IDs in the network-layer headers. In addition, packet sources use pseudonyms instead of their real IDs in onions so that onion nodes can not ascertain the initiators of received onions. Note that even the destination (an onion node as well) cannot determine who sends it the message if the source does not leak its true ID in the message for the destination. This means that the adversary can no longer directly ascertain packet sources and destinations. The adversary, however, may still be able to assign a probability to each node for being the source or destination of a given packet. In this section, we resides in investigate the resilience of AOS against such probabilistic attacks [1].

In the following, we first introduce two entropy-based metrics that we will use to measure source and destination anonymity. Then we will show how the random forwarding technique helps prevent internal attackers from finding out true onion nodes. Subsequently, we evaluate the capability of AOS providing source and destination anonymity under both the worst-case scenario and the more general scenario.

5.2.1 Anonymity metrics

In [7], Pfitzman and Hansen defined anonymity as "the state of being not identifiable within a set of subjects, the anonymity set," in which the anonymity set is "the set of all possible subjects." Later, anonymity set has long been a popular metric to measure the anonymity provided by anonymous communication systems. The larger the anonymity set, the better anonymity in general. Anonymity set unfortunately fails to reflect the possibly different probabilities assigned by the adversary to each node as being the source or destination of a packet. This issue was overcome by an entropy-based metric proposed in [37]. Let Ω denote a set of N nodes ($|\Omega| = N$) in an anonymous communication system. Briefly speaking, its anonymity entropy is defined by

$$\Upsilon = -\sum_{X \in \Omega} P_X \log_2(P_X)\,, \tag{9}$$

where P_X is the probability of node X being the source (or destination) of a packet assigned by the adversary after he observes the system. Υ measures the uncertainty that the adversary has about which node is the source or destination of a packet. One can also interpret Υ as the number of bits of additional information that the adversary needs to precisely identify the packet source (or destination). It follows that $0 \leq \Upsilon \leq \log_2(N)$ [38]. The lower bound is

achieved when source S (or destination D) is assigned a probability of one, while each $X \in \Omega \setminus \{S\}$ (or $X \in \Omega \setminus \{D\}$) is assigned a probability of zero; the upper bound is attained when each $X \in \Omega$ is assigned an equal probability of $1/N$, meaning that they are equal likely to be the source (or destination) as viewed by attackers (the ideal case). Further, Diaz et al. [39] defined degree of anonymity as

$$\upsilon = \frac{\Upsilon}{\log_2(N)}, \tag{10}$$

which measures how far the real anonymity entropy of a system is from the maximum anonymity entropy it can provide.

5.2.2 Efficacy of intra-clique random forwarding

For convenience, we call source clique $\mathcal{C}_s$ also an onion clique denoted by $\mathcal{A}_{0,1}$ and source S a real onion node denoted by $O_{0,1}$. It is critical in AOS to prevent the adversary from ascertaining real onion nodes $O_{i,1}$ $(0 \le i \le \alpha - 1)$. As discussed in Sect. 4.6, each onion $\mathcal{O}_{i+1}$ takes a random path starting from onion node $O_{i,1}$ before entering onion layer $i + 1$. This helps preventing global external attackers from identifying $O_{i,1}$. Unfortunately, there might be compromised nodes (internal attackers) on the forwarding path. Below we show the efficacy of the random-forwarding technique against such internal attackers.

Consider a set of $c \in [1, |\mathcal{A}_i| - 1]$[2] internal attackers in clique $\mathcal{A}_{i,1}$ $(0 \le i \le \alpha - 1)$, among which at least one appears on the forwarding path of onion $\mathcal{O}_{i+1}$. The internal attackers aim to determine which non-compromised node in $\mathcal{A}_{i,1}$ is $O_{i,1}$ that initiated $\mathcal{O}_{i+1}$. Let U be the first internal attacker which received $\mathcal{O}_{i+1}$ from a non-compromised node V. From the viewpoint of the internal attackers, all the non-compromised nodes in $\mathcal{A}_{i,1}$ other than V are each equally likely to be $O_{i,1}$, but they are also obviously less likely to be $O_{i,1}$ than V. We now analyze how confident the internal attackers can be that V is indeed $O_{i,1}$ or equivalently the probability that they assigned to V being $O_{i,1}$.

Theorem 1 *Assuming that the first internal attacker U in clique $\mathcal{A}_{i,1}, \forall i \in [0, \alpha - 1]$, received onion $\mathcal{O}_{i+1}$ from node V, the probability that the internal attackers assigned to V as the onion node $O_{i,1}$ is $\frac{n-(n-c-1)\eta}{n}$, where $n = |\mathcal{A}_{i,1}|$ and c is the number of internal attackers in clique $\mathcal{A}_{i,1}$.*

[2] The real onion node $O_{i,1}$ is not compromised.

Proof Denote by $\overrightarrow{VU}$ the event that U received onion $\mathcal{O}_{i+1}$ from node V. The probability of V being $O_{i,1}$ is

$$\mathbf{Pr}(V = O_{i,1} | \overrightarrow{VU}) = \frac{\mathbf{Pr}(V = O_{i,1}, \overrightarrow{VU})}{\mathbf{Pr}(\overrightarrow{VU})}, \tag{11}$$

where

$$\begin{aligned} \mathbf{Pr}(\overrightarrow{VU}) &= \mathbf{Pr}(V = O_{i,1}, \overrightarrow{VU}) + \mathbf{Pr}(V \neq O_{i,1}, \overrightarrow{VU}) \\ &= \mathbf{Pr}(V = O_{i,1}) \cdot \mathbf{Pr}(\overrightarrow{VU} | V = O_{i,1}) \\ &\quad + \mathbf{Pr}(V \neq O_{i,1}) \cdot \mathbf{Pr}(\overrightarrow{VU} | V \neq O_{i,1}). \end{aligned} \tag{12}$$

Denote by $Pos(U) = k$ the event that U occupies the kth position in the path. Then $p'_k = \mathbf{Pr}(\overrightarrow{VU}, Pos(U) = k | V = O_{i,1})$ means the probability that V is indeed $O_{i,1}$ and $\mathcal{O}_{i+1}$ passed $k - 1$ non-compromised nodes before V. There are three cases:

- $k = 1$: This means that V chose U as the first onion forwarder, which occurs with probability $p'_1 = \frac{1}{n}$ because each onion forwarder is chosen uniformly at random from clique $\mathcal{A}_{i,1}(|\mathcal{A}_{i,1}| = n)$.
- $k = 2$: This means that V chose itself as the first onion forwarder and then U as the second one. This occurs with probability $p'_2 = \frac{\eta}{n^2}$.
- $k > 2$: This means that onion $\mathcal{O}_{i+1}$ passed $k - 2$ non-compromised nodes, among which the last one chose V as the $(k - 1)$th onion forwarder which in turn selected U as the kth onion forwarder. This case happens with $p'_k = \eta^{k-1} \cdot (\frac{n-c}{n})^{k-2} \cdot \frac{1}{n^2} = \frac{\eta}{n^2}\left(\frac{(n-c)\eta}{n}\right)^{k-2}$. It follows that

$$\begin{aligned} \mathbf{Pr}(\overrightarrow{VU} | V = O_{i,1}) &= \frac{1}{n} + \frac{\eta}{n^2} \sum_{j=0}^{\infty} \left(\frac{(n-c)\eta}{n}\right)^j \\ &= \frac{n - (n-c-1)\eta}{n^2 - n(n-c)\eta}. \end{aligned} \tag{13}$$

Similarly, we have

$$\begin{aligned} \mathbf{Pr}(\overrightarrow{VU} | V \neq O_{i,1}) &= \sum_{k=1}^{\infty} \mathbf{Pr}(\overrightarrow{VU}, Pos(U) = k | V \neq O_{i,1}) \\ &= 0 + \sum_{k=2}^{\infty} \eta^{k-1} \cdot \left(\frac{n-c}{n}\right)^{k-2} \cdot \frac{1}{n^2} \\ &= \frac{\eta}{n^2 - n(n-c)\eta}. \end{aligned} \tag{14}$$

Finally, if U has no other information, all the non-compromised nodes in clique $\mathcal{A}_{i,1}$ are each equally probable to be the onion node $O_{i,1}$. So we have

$$\mathbf{Pr}(V = O_{i,1}) = \frac{1}{n-c} \quad \text{and} \quad \mathbf{Pr}(V \neq O_{i,1}) = \frac{n-c-1}{n-c}. \tag{15}$$

Substituting Eq. (13), Eq. (14), and Eq. (15), into Eq. (12) and then Eq. (11), we finally get

$$\mathbf{Pr}(V = O_{i,1}|\overrightarrow{VU}) = \frac{n-(n-c-1)\eta}{n}. \tag{16}$$

□

From Eq. (16), we can see that

- When $n = c + 1$, which means that all the nodes in $\mathcal{A}_{i,1}$ other than V are compromised, $\mathbf{Pr}(V = O_{i,1}|\overrightarrow{VU}) = 1$, so the adversary can ascertain that V is indeed $O_{i,1}$.
- When $n > c + 1$, the larger η, the smaller $\mathbf{Pr}(V = O_{i,1}|\overrightarrow{VU})$, the better the onion node is hidden from the internal attackers, and the larger the communication overhead totalTran (cf. Eq. (8)).

Also note that all the other $n - c - 1$ non-compromised nodes equally share the residue probability of being $O_{i,1}$. In summary, the probability distribution of each node in $\mathcal{A}_{i,1}$ being $O_{i,1}$ as viewed by the adversary is given by

$$\mathbf{Pr}(X = O_{i,1}) = \begin{cases} \frac{n-(n-c-1)\eta}{n} & X = V \\ \frac{\eta}{n} & X \in \mathcal{A}_{i,1} \text{ and } X \neq V \\ 0 & \text{else}. \end{cases} \tag{17}$$

5.2.3 Anonymity measurement: the worst-case scenario

We first consider the worst scenario in which there is only one communication session from source S to destination D in the whole network. The adversary with global eavesdropping capability can thus easily differentiate between inter-clique and intra-clique packets and identify the source clique $\mathcal{C}_s$ and all the onion cliques $\mathcal{A}_{i,j}, \forall i \in [1, \alpha], j \in [1, \beta_i]$. We also assume that there is at least one internal attacker acting as the onion forwarder for each onion $\mathcal{O}_i$ so that all the β onion nodes are known to the adversary.

Let us examine the source anonymity. We just need to consider the first internal attacker in clique $\mathcal{C}_s$ participating in forwarding $\mathcal{O}_1$ because it is in the best position to identify source S. The probability of every node being the source as viewed by the adversary follows the distribution in Eq. (17), based on which we can derive the source-anonymity entropy using Eq. (9) and the source-anonymity degree using Eq. (10).

To evaluate the destination anonymity, we assume that the adversary compromised b out of the β onion nodes, but not yet finding the destination. Therefore, the remaining $\beta - b$ onion nodes, denoted by $\mathcal{B}_{\beta-b}$, are each equally likely to be the destination with probability $1/(\beta - b)$. The probability of every network node being the destination as viewed by the adversary thus follows the distribution

$$\mathbf{Pr}(X = D) = \begin{cases} \frac{1}{\beta-b} & \text{if } X \in \mathcal{B}_{\beta-b}, \\ 0 & \text{else}. \end{cases} \tag{18}$$

Then we can derive the destination-anonymity entropy using Eq. (9) and the destination-anonymity degree using Eq. (10).

5.2.4 Anonymity measurement: the general scenario

In more general cases, there would be multiple concurrent communication sessions in the network involving different source-destination pairs. To enable theoretical analysis, we assume that the network traffic is *active* and *uniform* in the sense that there are both incoming and outgoing traffic at every clique as a whole within every sufficiently small time interval. It is, therefore, infeasible for the adversary to precisely differentiate communication sessions or, equivalently identify all the cliques associated with every session via timing analysis or other external attacks [1]. This allows us to focus on the impact of internal attackers. Without loss of generality, we still consider the session from source S to destination D.

Let us first consider the impact of consecutively compromised real onion nodes. Assume that the adversary compromised C out of the overall N nodes. Due to the layered encryption, the same message appears entirely different across onion layers so that only real onion nodes $O_{i,1}(1 \leq i \leq \alpha - 1)$ that peel off one layer of encryption can link messages across two adjacent layers. Note that the last real onion node $O_{\alpha,1}$ is excluded here because it does not further forward the onion. We refer to a chain of a $(1 \leq a \leq \alpha - 1)$ compromised real onion nodes serving the same communication session as an *a-chain*. Each a-chain can link the same message across $a + 1$ consecutive onion layers. In particular, if $a = 1$, there are no consecutively compromised real onion nodes; if $a = \alpha - 1$, then the adversary can trace the message from clique $A_{1,1}$ to $A_{\alpha,1}$. There might be multiple disjoint chains, but the adversary cannot correlate them together. So we just need to consider the longest a-chain, called the a^*-chain, which reveals the most information the adversary. The a^*-chain can start at any of the first $\alpha - a^*$ real onion nodes, and consequently, the probability of it starting from $O_{1,1}$ is simply $1/(\alpha - a^*)$. Or equivalently, assuming that the a^*-chain starts from $O_{i,1}(1 \leq i \leq \alpha - 1)$ and that there is at least one internal attacker in clique $\mathcal{A}_{i-1,1}$ participating in random onion forwarding, $\mathcal{A}_{i-1,1}$ is the source clique $\mathcal{A}_{0,1} = \mathcal{C}_s$ with probability $1/(\alpha - a^*)$.

To enable the quantitative analysis, we also consider an extreme case in which there is at least one internal attacker

participating in random onion forwarding in each onion clique $\mathcal{A}_{i,1}, 0 \le i \le \alpha - 1$. For simplicity, we also assume that the same parameter β is used in all communication sessions, that each onion layer i has the same number $\beta_i = \beta/\alpha$ of onion nodes, that the adversary knows both β and α, and that each clique contains the same number n of nodes (i.e., $N = Mn$). Note that all these assumptions are in favor of the adversary. For example, it is possible that no a-chains exist, in which case the anonymity of S and D will be certainly better than what we will evaluate below.

We have the following theorem about the source anonymity.

Theorem 2 *Assuming that the a^*-chain starting from $O_{i,1}(i \in [1, \alpha])$ and that the first internal attacker U in clique $\mathcal{A}_{i-1,1}$ received onion $\mathcal{O}_i$ from node V, the probability that the adversary assigned to V as the source is $\frac{n-(n-c-1)\eta}{n(\alpha-a^*)}$, where c is the number of compromised nodes in $\mathcal{A}_{i-1,1}$.*

Proof The proof follows immediately that of Theorem 1. Since the probability of U in source clique $\mathcal{C}_s$ is $1/(\alpha - a)$, the probability of U's immediate predecessor V in the random-forwarding path being source S is then given by

$$\begin{aligned}\mathbf{Pr}(V = S|\overrightarrow{VU}) &= \mathbf{Pr}(V = S|\overrightarrow{VU}, U \in \mathcal{C}_s) \cdot \mathbf{Pr}(U \in \mathcal{C}_s) \\ &= \frac{n-(n-c-1)\eta}{n} \cdot \frac{1}{(\alpha - a^*)} \\ &= \frac{n-(n-c-1)\eta}{(\alpha - a^*)n}. \end{aligned} \tag{19}$$

□

Similarly, every non-compromised nodes other than V in $\mathcal{A}_{i-1,1}$ has the equal probability $\frac{\eta}{(\alpha-a^*)n}$ of being the source. Assuming that there are C compromised nodes (not including source S and destination D) among all the N nodes in the network, all the other $(M-1)n - (C-c)$ non-compromised nodes not in $\mathcal{A}_{i-1,1}$ equally share the residue probability $\frac{\alpha-a^*-1}{\alpha-a^*}$ of being the source. In summary, the probability distribution of every node being the source as viewed by the adversary is given by

$$\mathbf{Pr}(X = S) = \begin{cases} \frac{n-(n-c-1)\eta}{(\alpha-a^*)n} & X = V \\ \frac{\eta}{(\alpha-a^*)n} & X \in \mathcal{A}_{i-1,1}, X \neq V \\ \frac{\alpha-a-1}{(\alpha-a^*)(Mn-n-C+c)} & \text{else}. \end{cases} \tag{20}$$

Then we can derive the source-anonymity entropy using Eq. (9) and the source-anonymity degree using Eq. (10).

The following theorem is about the destination anonymity.

Theorem 3 *Assuming that the $a^* + 1$ onion layers linked by the a^*-chain in the path contain β' onion nodes, among which b are compromised, then the probability of being the destination assigned by the adversary to each of the $\beta' - b$ remaining non-compromised onion nodes is $1/(\beta - b)$.*

Proof It is obvious that $b \in [a^*, \beta']$. Since the adversary knows that there are β onion nodes in total and b of them cannot be the destination, then every non-compromised onion node, if known, has the same probability $1/(\beta - b)$ of being the destination. □

Since the adversary cannot identify the remaining $\beta - \beta'$ onion nodes from the $N - C - (\beta' - b)$ non-compromised nodes (denoted by Ψ), all these nodes equally share the probability of $\frac{\beta-\beta'}{\beta-b}$ of being an onion node or equivalently the destination. Therefore, the probability distribution of every node being the destination as viewed by the adversary is

$$\mathbf{Pr}(X = D) = \begin{cases} \frac{1}{\beta-b} & X \in \mathcal{B}_{\beta'-b} \\ \frac{\beta-\beta'}{(\beta-b)(N-C-\beta'+b)} & \Psi \\ 0 & else. \end{cases} \tag{21}$$

Similarly, we can derive the destination-anonymity entropy and degree using Eq. (9) and using Eq. (10), respectively.

5.3 Numerical results

In this subsection, we use concrete numerical results to demonstrate the effectiveness of AOS as well as the relationship between source/destination anonymity and multiple related parameters.

5.3.1 The probability distribution of a^*

As shown in Sect. 5.2, source anonymity is largely related to the a^*-chain or $\alpha - a^*$ (cf. Eq. (20)). Table 2 shows the probability distribution of a^* under different value of α, where we assume that each node is compromised independently with probability 0.1, which is considered a severe situation. Due to the space limitation, we omit the straightforward calculation details.

As shown in Table 2, larger α can lead to larger $\alpha - a^*$ with overwhelming probability. For example, when $\alpha =$

Table 2 Probability distribution of a^* vs. α

α	The probability of $a^* = X$					
	$X = 0$	$X = 1$	$X = 2$	$X = 3$	$X = 4$	$X = 5$
2	0.8100	0.1900	0	0	0	0
3	0.7290	0.2520	0.0190	0	0	0
4	0.6561	0.3159	0.0261	0.0019	0	0
5	0.5905	0.3726	0.0341	0.0026	0.0002	0
6	0.5314	0.4228	0.0420	0.0034	0.0003	$<10^{-4}$

$4, \alpha - a^* \geq 3$ with probability 0.972. Note that destination anonymity also depends on a^* because the larger a^*, the more candidate destinations the adversary might know (cf. Eq. (21)).

5.3.2 Source anonymity

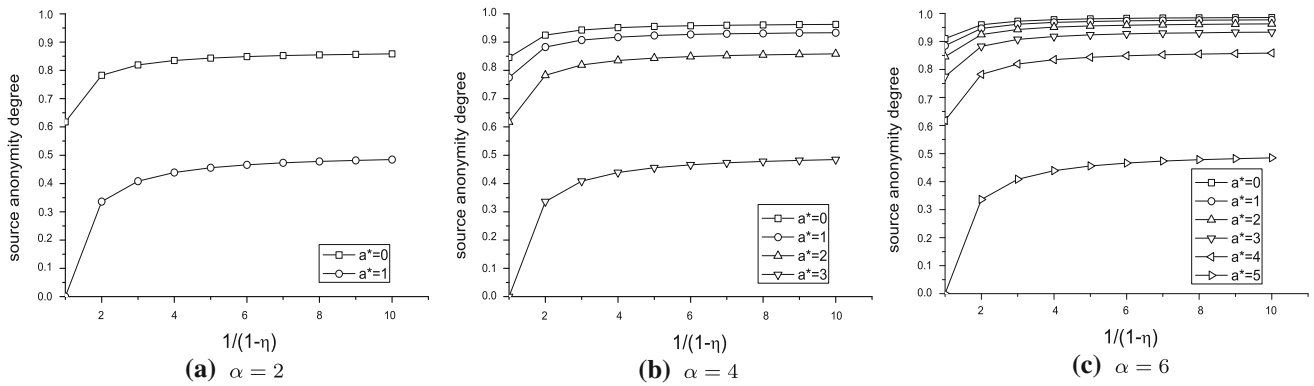
Figure 2 shows the source-anonymity degree varying with η, α, and a^* (cf. Eq. (20)), where the network size $N = 400$, the number of cliques $M = 20$, and $C = 40$ nodes are compromised. Since the expected number of hops in intra-clique random forwarding is $1/(1-\eta)$ (cf. Sect. 5.1), we use $1/(1-\eta)$ instead of η as the x-axis to demonstrate the relationship between communication overhead (cf. Eq. (8)) and source anonymity.

As we can see, besides the fact that increasing α can effectively enhance source anonymity, an average of $3 \sim 4$ hops of random forwarding can also lead to a significant increase in source anonymity. Also note that with $\eta > 0$, it is impossible for the adversary to pinpoint the source even when the first $\alpha - 1$ real onion nodes are all compromised. These results demonstrate the efficacy of intra-clique random forwarding in improving source anonymity.

5.3.3 Destination anonymity

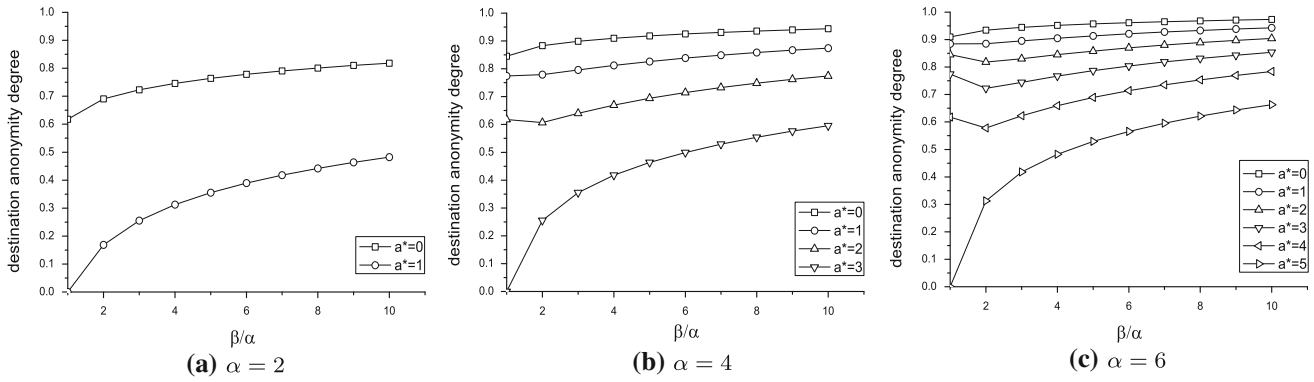
Figure 3 shows the destination-anonymity degree (cf. Eq. (21)), where again $N = 400$, $M = 20$, and $C = 40$. For simplicity, we assume that $\beta_i = \beta/\alpha, \forall i \in [1, \alpha]$, i.e., the same number of onion nodes at each onion layer. In practice, however, β_i should be chosen randomly to prevent the adversary from linking packets across different onion layers.

We can see that increasing α and/or β can generally enhance destination anonymity. Generally speaking, with $\beta_i \geq 1$, it is much more difficult for the adversary to precisely identify the destination, as it has to compromise all the other $\beta - 1$ onion nodes. One may notice that increasing α may occasionally reduce destination anonymity, e.g., when $\alpha = 4, a^* = 2$, and β_i increases from 1 to 2. This can be explained as follows. Referring to the measurement of destination anonymity in Sect. 5.2.4, when $\beta_i = 1$, the adversary knows $(a^* + 1)\beta_i = 3$ onion nodes, among which $a^* = 2$ have been compromised and are not the destination. So there is only one candidate destination with probability $1/(\beta - a^*) = 0.5$ being the destination, and all the other non-compromised nodes equally share the residue probability 0.5 of being the destination. When



(a) $\alpha = 2$ **(b)** $\alpha = 4$ **(c)** $\alpha = 6$

Fig. 2 Source anonymity vs. $\alpha/\eta/a^*$



(a) $\alpha = 2$ **(b)** $\alpha = 4$ **(c)** $\alpha = 6$

Fig. 3 Destination anonymity vs. $\alpha/\beta/a^*$

$\beta_i = 2$, the adversary knows $(a^* + 1)\beta_i = 6$ onion nodes, among which $a^* = 2$ have been compromised and are not the destination. So there are four candidate destinations with probability $1/(\beta - a^*) = 1/6$ being the destination, and all the other non-compromised nodes equally share the residue probability 1/3 of being the destination. The destination-anonymity entropy of the latter case is slightly less than that of the first case, so does the destination anonymity degree.

5.3.4 The impact of the number of compromised nodes

Figure 4 illustrates the impact of the number of compromised nodes, where $N = 400$, $M = 20$, $\alpha = 4$, $\eta = 0.75$, and $\beta_i = 2$. As we can see, the more compromised nodes, the lower source and destination anonymity, which coincides with the intuition. In particular, when $c = 200$, i.e., half of the nodes are compromised, the source and destination anonymity degrees are still higher than 0.9 and 0.75, respectively. The results demonstrate that AOS is resilient to node compromise.

5.3.5 The impact of the clique size

The source anonymity is also related to the clique size n (cf. Eq. (20)). Figure 5 shows the source anonymity varying with the clique size, where $N = 1000$ and $C = 100$. It is obvious that a larger clique size will lead to higher source anonymity.

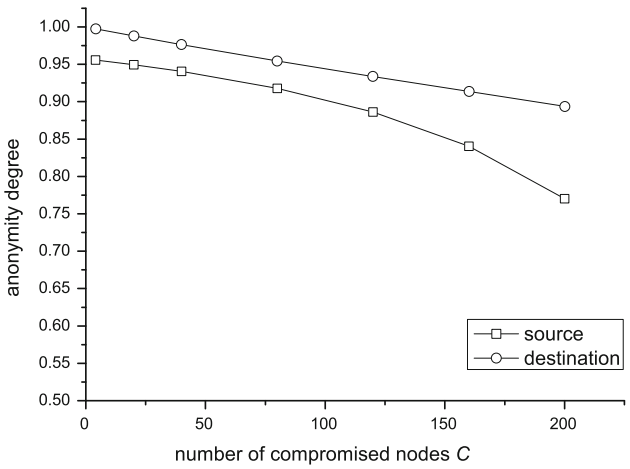


Fig. 4 Source/destination anonymity vs. the number of compromised nodes

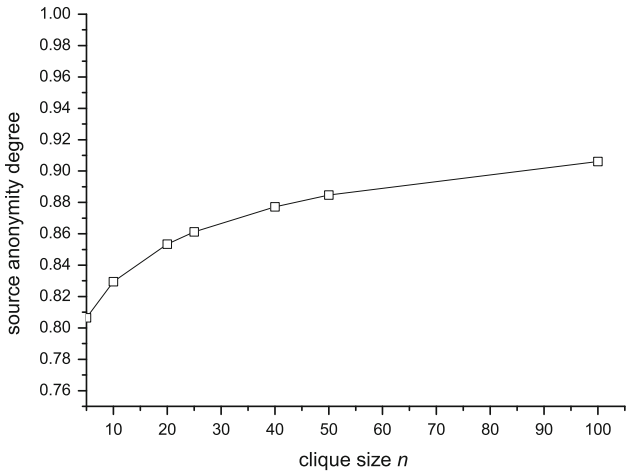


Fig. 5 Source anonymity vs. the clique size

5.4 Discussion

5.4.1 The choice of parameters

As shown in Sect. 5.3, AOS has a number of parameters that influence not only source and destination anonymity but also the communication overhead (including onion Overhead and totalTran). We discuss the choice of these parameters here.

- The larger α, the higher source and destination anonymity, the fewer concurrent sessions can be supported, and vice versa. In practice, α can be chosen conservatively, i.e., $\alpha = 3$ or 4.
- The larger η, the higher source anonymity, the fewer concurrent sessions can be supported, and vice versa. The gain from increasing η is most significant when $1/(1 - \eta) = 3$ or 4, which corresponds to $\eta \approx 0.67$ or 0.75, respectively.
- The larger β, the higher destination anonymity, the larger the communication overhead, and vice versa. To provide differentiated destination anonymity, we can use larger βs for nodes with higher anonymity requirements and smaller βs for those with lower anonymity requirements.
- The larger λ, the larger the communication overhead, the more concurrent sessions can be supported, and vice versa.
- The larger the clique size, the higher source anonymity, the heavier the total intra-clique cover traffic (larger communication overhead), and vice versa. To provide differentiated source anonymity, we can hide VIP nodes with higher anonymity requirements in larger cliques while letting most cliques be of smaller sizes.

5.4.2 The placement of the destination

In general, the destination should be placed at a randomly chosen onion layer such that adversary cannot get any empirical information. Otherwise, for example, if the destination is placed near the source for the majority of sessions, the adversary might directly rule out some candidates in the far end of an a-chain. For fewer time-critical sessions, however, the source can intentionally place the destination at the first onion layer to minimize the latency. So AOS also provides differentiated communication latency.

6 Conclusion

In this paper, we presented the design and evaluation of AOS, a novel anonymous overlay system for MANETs. In contrast to previous research, AOS is independent of the underlying MANET protocol stack, can coexist with indispensable secure MANET routing schemes, and can provide differentiated anonymity protection to MANET nodes with diverse anonymity requirements. The efficacy of AOS in offering strong source and destination anonymity has been theoretically proved and thoroughly evaluated via numerical results. As the future work, we intend to evaluate AOS using network simulations and experiments. We will also seek to analyze the security of AOS under other adversary models.

Acknowledgments The work of Y. Zhang was partially supported by the US National Science Foundation under grants CNS-0716302 and CNS-0844972. The work of Y. Fang was partially supported by the US National Science Foundation under grant CNS-0716450, the National Natural Science Foundation of China under grant 61003300, and China 111 Project under grant B08038.

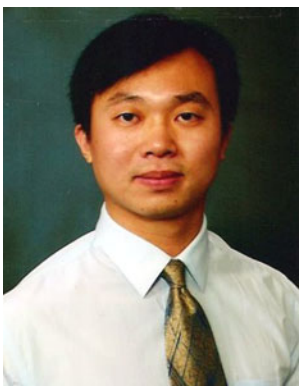
References

1. Raymond, J.-F. (2000). Traffic analysis: Protocols, attacks, design issues, and open problems. In: *International workshop on design issues in anonymity and unobservability* (pp. 10–29). Berkeley, CA.
2. Jiang, S., Vaidya, N., & Zhao, W. (2001). Prevent traffic analysis in packet radio networks. In: *Proceedings of DISCEX II*, Anaheim, California.
3. Defense Advanced Research Projects Agency (DARPA). (1998). Research challenges in high confidence networking, White paper, Arlington, VA, July 1998.
4. Chaum, D. (1981). Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM, 24*(2), 84–90.
5. Pfitzmann, A., & Waidner, M. (1987). Networks without user observability. *Computers & Security, 6*(2), 158–166.
6. Reiter, M., & Rubin, A. (1998). Crowds: Anonymity for web transactions. *ACM TISSEC, 1*(1):66–92.
7. Pfitzmann, A., & Hansen, M. (2005). Anonymity, unobservability, and pseudonymity: A consolidated proposal for terminology. Draft v0.25, Dec. 2005.
8. Kong, J., & Hong, X. (2003). ANODR: Anonymous on demand routing with untraceable routes for mobile ad-hoc networks. In: *ACM MobiHoc'03* (pp. 291 – 302). Annapolis, MD.
9. Jiang, S., Vaidya, N. H., & Zhao, W. (2004). A mix route algorithm for mix-net in wireless mobile ad hoc networks. In: *MASS'04* (pp. 406–415). Fort Lauderdale, FL.
10. Zhu, B., Wan, Z., Kankanhalli, M. S., Bao, F., & Deng, R. H. (2004). Anonymous secure routing in mobile ad-hoc networks. In *LCN'04* (pp. 102–108). Dublin, Ireland.
11. Wu, X., & Bhargava, B. (2005). AO2P: Ad hoc on-demand position-based private routing protocol. *4*(4), 335–348.
12. Zhang, Y., Liu, W., & Lou, W. (2005). Anonymous communications in mobile ad hoc networks. In *IEEE INFOCOM'05* (pp. 1940–1951). Miami, FL.
13. Zhang, Y., Liu, W., Lou, W., & Fang, Y. (2006). MASK: Anonymous on-demand routing in mobile ad hoc networks. *IEEE Transactions On Wireless Communications, 5*(9), 2376–2385.
14. Choi, H., McDaniel, P., & La Porta, T. F. (2007). Privacy preserving communication in MANETs. In *IEEE SECON'07* (pp. 233–242). San Diego, CA.
15. Aad, I., Castelluccia, C., & Hubaux, J.-P. (2006). Packet coding for strong anonymity in ad hoc networks. In *SecureComm'06*, Baltimore, MD.
16. Chou, C.-C., Wei, D. S., Kuo, C.-C. J., & Naik, K. (2007). An efficient anonymous communication protocol for peer-to-peer applications over mobile ad-hoc networks. *IEEE Journal on Selected Areas in Communications, 25*(1), 192–203.
17. Dong, Y., Chim, T. W., Li, V. O., Yiu, S., & Hui, C. (2009). ARMR: Anonymous routing protocol with multiple routes for communications in mobile ad hoc networks. *Ad Hoc Networks, 7*(8), 1536–1550.
18. El Defrawy, K., & Tsudik, G. (2007). ALARM: Anonymous location-aided routing in suspicious MANETs. In: *ICNP'07* (pp. 304–313). Beijing, China.
19. Kao, J.-C., & Marculescu, R. (2007). Real-time anonymous routing for mobile ad hoc networks. In *WCNC'07* (pp. 4139–4144). Hongkong, China.
20. El Defrawy, K., & Tsudik, G. (2008). PRISM: Privacy-friendly routing in suspicious MANETs (and VANETs). In *ICNP'08* (pp. 258–267). Orlando, FL.
21. Wu, X., Liu, J., Hong, X., & Bertino, E. (2008). Anonymous geo-forwarding in MANETs through location cloaking. *IEEE Transactions on Parallel and Distributed Systems, 19*(10), 1297–1309.
22. Hu, Y.-C., Perrig, A., & Johnson, D. B. (1986). Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In *IEEE INFOCOM'03* (pp. 1976–1986). San Francisco, CA.
23. Hu, Y.-C., Perrig, A., & Johnson, D. B. (2003). Rushing attacks and defense in wireless ad hoc network routing protocols. In *WiSe'03* (pp. 30–40). San Diego, CA.
24. Hu, Y.-C., Perrig, A., & Johnson, D. B. (2002). Ariadne: A secure on-demand routing protocol for ad hoc networks. In: *ACM MobiCom'02* (pp. 12–23). Atlanta, GA.
25. Hu, Y.-C., Johnson, D. B., & Perrig, A. (2003). SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks. *Ad Hoc Networks, 1*(1), 175–192.
26. Sanzgiri, K., LaFlamme, D., Dahill, B., Levine, B., Shields, C., & Belding-Royer, E. (2005). Authenticated routing for ad hoc networks. *IEEE Journal on Selected Areas in Communications, 23*(3), 598–610.
27. Reed, M., Syverson, P., & Goldschlag, D. (1998). Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications, 16*(4), 482–494.
28. Camenisch J., & Lysyanskaya, A. (2005). A formal treatment of onion routing. In: *Advances in Cryptology—CRYPTO 2005* (pp. 169–187). Santa Barbara, California, USA.
29. Zhang, Y., Lou, W., & Fang, Y. (2007). A secure incentive protocol for mobile ad hoc networks. *Wireless Networks, 13*(5): 569–582.

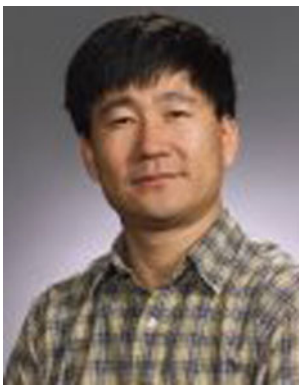
30. Zhang, Y., Liu, W., Lou, W., & Fang, Y. (2006). Securing mobile ad hoc networks with certificateless public keys. *IEEE Transactions on Dependable and Secure Computing, 3*(4), 386–399
31. Boneh, D., & Franklin, M. (2001). Identity-based encryption from the weil pairing. In: *CRYPTO'01, Santa Barbara* (pp. 213–229). CA.
32. Barreto, P., Kim, H., Bynn, B., & Scott, M. (2002). Efficient algorithms for pairing-based cryptosystems. In *CRYPTO'02* (pp. 354–368). Santa Barbara, CA.
33. Kate, A., Zaverucha, G., & Goldberg, I. (2007). Pairing-based onion routing. In *PETS'07*, Ottawa, Canada.
34. Wright, M. K., Adler, M., Levine, B. N., & Shields, C. (2004). The predecessor attack: An analysis of a threat to anonymous communications systems. *ACM Transactions on Information and System Security, 7*(4), 489–522.
35. Danezis, G., Diaz, C., Kasper, E., & Troncoso, C. (2009). The wisdom of Crowds: Attacks and optimal constructions. In *ESORICS'09*, St Malo, France.
36. Scott, M. (2005). Computing the tate pairing. In *CT-RSA'05* (pp. 293–304). San Francisco, CA.
37. Serjantov, A., & Danezis, G. (2002). Towards an information theoretic metric for anonymity. In *PET'02*, ser. LNCS vol. 2482, (pp. 41–53). Berlin: Springer.
38. Cover, T. M., & Thomas, J. A. (2006). *Elements of information theory* (2nd edn.). London: Wiley.
39. Díaz C., Seys, S., Claessens, J., & Preneel, B. (2002). Towards measuring anonymity. In: *PET'02*, ser. LNCS, vol. 2482. (pp. 54–68). Berlin: Springer.

Author Biographies

Rui Zhang is a Ph.D. candidate in School of Electrical, Computer, and Engineering at Arizona State University. He received the B.E. degree in Communication Engineering and the M.E. degree in Communication and Information System from Huazhong University of Science and Technology, China, in 2001 and 2005, respectively. Before starting his Ph.D. studies in 2008, he was a software engineering in UTStarcom Shenzhen R&D center between 2005 and 2007.



Yanchao Zhang received the B.E. in Computer Science & Technology from Nanjing University of Posts & Telecom, China, in 1999, the M.E. in Computer Science & Technology from Beijing University of Posts & Telecom, China, in 2002, and the Ph.D. in Electrical and Computer Engineering from the University of Florida in 2006. He joined Arizona State University in June 2010 as an Associate Professor of School of Electrical, Computer, and Energy Engineering. Before coming to ASU, he was an Assistant Professor of Electrical and Computer Engineering at New Jersey Institute of Technology from 2006 to 2010. His primary research interests are network and distributed system security, wireless networking, and mobile computing. He is an Associate Editor of IEEE Transactions on Vehicular Technology, a Feature Editor of IEEE Wireless Communications, and a Guest Editor of IEEE Wireless Communications Special Issue on Security and Privacy in Emerging Wireless Networks. He is a TPC Co-Chair of Communication and Information System Security Symposium, IEEE GLOBECOM 2010, and a Workshop Co-Chair of ACM MSWiM'10. He received the NSF CAREER Award in 2009.



Yuguang Fang (F'08) received a Ph.D. degree in Systems Engineering from Case Western Reserve University in January 1994 and a Ph.D degree in Electrical Engineering from Boston University in May 1997. He was an assistant professor in the Department of Electrical and Computer Engineering at New Jersey Institute of Technology from July 1998 to May 2000. He then joined the Department of Electrical and Computer Engineering at University of Florida in May 2000 as an assistant professor, got an early promotion to an associate professor with tenure in August 2003 and to a full professor in August 2005. He holds a University of Florida Research Foundation (UFRF) Professorship from 2006 to 2009, a Changjiang Scholar Chair Professorship with Xidian University, Xi'an, China, from 2008 to 2011, and a Guest Chair Professorship with Tsinghua University, China, from 2009 to 2012. He has published over 250 papers in refereed professional journals and conferences. Dr. Fang received the National Science Foundation Faculty Early Career Award in 2001 and the Office of Naval Research Young Investigator Award in 2002, and is the recipient of the Best Paper Award in IEEE International Conference on Network Protocols (ICNP) in 2006 and the recipient of the IEEE TCGN Best Paper Award in the IEEE High-Speed Networks Symposium, IEEE Globecom in 2002. Dr. Fang is also active in professional activities. He is a Fellow of IEEE and a member of ACM. He is currently serving as the Editor-in-Chief for IEEE Wireless Communications and serves/served on several editorial boards of technical journals including IEEE Transactions on Communications, IEEE Transactions on Wireless Communications, IEEE Wireless Communications Magazine and ACM Wireless Networks. He was an editor for IEEE Transactions on Mobile Computing and currently serves on its Steering Committee. He has been actively participating in professional conference organizations such as serving as the Steering Committee Co-Chair for QShine from 2004 to 2008, the Technical Program Vice-Chair for IEEE INFOCOM'2005, Technical Program Symposium Co-Chair for IEEE Globecom'2004, and a member of Technical Program Committee for IEEE INFOCOM (1998, 2000, 2003–2010).