ANOC: Anonymous Network-Coding-Based Communication with Efficient Cooperation

Peng Zhang, Chuang Lin, Yixin Jiang, Patrick P.C. Lee, and John C.S. Lui

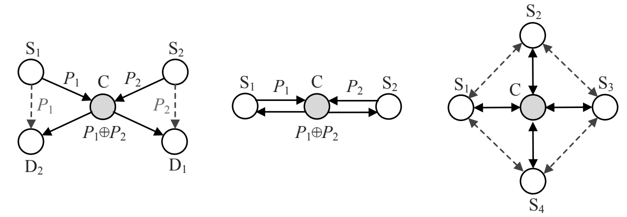
***Abstract*—Practical wireless network coding (e.g., COPE) is a promising technique that can enhance the throughput of wireless networks. However, such a technique also bears a serious security drawback: it breaks the current privacy-preserving protocols (e.g., Onion Routing), since their operations conflict each other. As user privacy in wireless networks is highly valued nowadays, a new privacy-preserving scheme that can function with wireless network coding becomes indispensable.**

To address such a challenge, we apply the idea of *cooperative networking* and design a novel anonymity scheme named ANOC, which can function in network-coding-based wireless mesh networks. ANOC is built upon the classic Onion Routing protocol, and resolves its conflict with network coding by introducing efficient cooperation among relay nodes. Using ANOC, we can perform network coding to achieve a higher throughput, while still preserving user privacy in wireless mesh networks. We formally show how ANOC achieves the property of *relationship anonymity*, and conduct extensive experiments via nsclick to demonstrates its feasibility and efficiency when integrated with network coding.

***Index Terms*—Network coding, anonymity, cooperative networking, Onion Routing.**

I. Introduction

HOW to achieve high data throughput is a critical concern in wireless networks. Recent studies show that network coding [1], as an alternative to the traditional store-and-forward paradigm, can remarkably enhance the network capacity. In particular, authors in [2] propose COPE, the first practical wireless network coding scheme for wireless mesh networks [3]. In COPE, nodes operate in promiscuous mode, and opportunistically perform data mixing (or coding) on the packets to be forwarded to neighboring nodes. Fig. 1 shows three basic coding scenarios in COPE [4]. In Fig. 1(a), node S_1 needs to send a packet P_1 to D_1, and this packet is relayed by node C; while S_2 needs to send a packet P_2 to D_2, also relayed by node C. The dashed line means that D_1 and D_2 can *overhear* P_2 and P_1, respectively, due to the broadcast nature of wireless channels. Without network coding, the communication will cost four transmissions in total: (1) S_1 sends P_1 to C, (2) C forwards P_1 to D_1, (3) S_2 sends P_2 to C, and (4) C forwards P_2 to D_2. On the other hand, with network coding, the relay node C only needs to broadcast $P_1 \oplus P_2$, and then D_1 can recover P_1 by computing $P_2 \oplus (P_1 \oplus P_2)$; D_2 can recover P_2 by computing $P_1 \oplus (P_1 \oplus P_2)$. In this way, one transmission will be saved at node C, and the network throughput can be improved. Fig. 1(b) shows another possible coding scenario where no overhearing is needed; Fig. 1(c) gives a hybrid scenario that combines the former two cases.



(a) Coding scenario with overhearing

(b) Coding scenario without overhearing

(c) Hybrid coding scenario

Fig. 1. Basic coding scenarios of COPE [4].

In addition to throughput improvement, privacy preservation is also an important concern in wireless communications since: (1) online privacy is highly valued by wireless users nowadays; and (2) the open-air traffic in wireless medium can be easily monitored and traced. Consider for example, a scenario where multiple clients can access a server S via a wireless mesh network. Equipped with targeted antennas, an adversary can easily intercept traffic by staying close to server S, and then perform traffic analysis [5] so as to deduce the identities of users who have accessed S. Depending on the specific service provided by S, sensitive information, such as "who has accessed a web page or downloaded a file", will be disclosed. It is important to note that end-to-end encryption (e.g., SSL/TLS) only provides a limited form of privacy: while end-to-end encryption hides application payload from the adversary, the adversary can still learn the IP addresses of the client and the server in a data session.

Many techniques are proposed to provide user privacy in communication networks: Mix-Net [6], [7], [8], Onion Routing [9], [10], [11], and Crowds [12] are shown to be effective in wired networks; ANODR [13], WAR [14], and Onion Ring [15] are more suitable for wireless applications (please refer to [16] for a detailed related work). However, when a wireless network is upgraded to enable network coding, many of the above privacy-preserving protocols will not be applicable. The core reason is that the packet-mixing operations required by

Manuscript received 15 February 2011; revised 17 July 2011. This work is supported by the National Basic Research Program of China (No. 2010CB328105, 2011CB302703, 2009CB320504), the National Natural Science Foundation of China (No. 60932003, 60970101), the NSFC A3 Program (No. 61161140320), the RGC's AoE/E-02/08, and RGC 2150634. The work of Patrick P. C. Lee is supported in part by grant GRF 413910 from the Research Grant Council of Hong Kong.

P. Zhang and C. Lin are with the Department of Computer Science and Technology, Tsinghua University, Beijing, China (e-mail: {pzhang, yxjiang, clin}@csnet1.cs.tsinghua.edu.cn).

Y. Jiang is with the EPRI, China Southern Power Grid Co. Ltd., Guangzhou, China (e-mail: yxjiang@csnet1.cs.tsinghua.edu.cn).

P. P.C. Lee and J. C.S. Lui are with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong (e-mail: {pclee, cslui}@cse.cuhk.edu.hk).

Digital Object Identifier 10.1109/JSAC.2012.121018.

network coding are in *conflict* with the encryption/decryption operations required by the privacy-preserving schemes at relay nodes (details will be given in Section III). Considering the rising privacy concern, as well as the increasing bandwidth demand in wireless networks, an efficient privacy-preserving scheme that can work with wireless network coding becomes highly important.

To address the above issue, this paper proposes ANOC, i.e., Anoymous NetwOrk-Coding-based communication for wireless mesh networks. ANOC uses Onion Routing as its building block, and resolves the conflict between Onion Routing and network coding by introducing efficient cooperation (*session-key sharing* and *auxiliary decrypting*) among relay nodes. Specifically, we mainly address the following two challenges: (i) how to trigger the session-key sharing in an on-demand fashion, and (ii) how to efficiently and securely share session keys with neighbors without leaking any information to adversaries.

With these challenges addressed, we formally show that ANOC can achieve a practical privacy requirement called *relationship anonymity* (i.e., *unlinkability* [17]), meaning that adversaries cannot associate any sender with the corresponding receiver of a data session by simply observing the wireless traffic. We also conduct extensive experiments via nsclick [18] to show that ANOC can work efficiently with network coding in wireless mesh networks.

In summary, our contribution is two-fold: 1) to the best of our knowledge, this is the first paper to address the privacy vulnerability of wireless network coding; 2) we propose, implement, and evaluate a novel anonymous communication scheme for network-coding-based wireless mesh networks, using techniques of cooperative networking.

The remainder of this paper is organized as follows. Section II gives a formal statement of the problem to be studied. Section III motivates the basic idea of ANOC, the implementation of which is detailed in Section IV. Section V and Section VI present analytical and experimental results, respectively. Finally, Section VII concludes the paper.

II. Problem Statement

A. System Model

We consider a typical wireless mesh network [3] consisting of wireless routers and clients, as shown in Fig. 2. The routers have minimal mobility and form the infrastructure for clients. Some of these routers along the boundary of the network, termed *proxy routers*, are responsible for setting up routes for clients directly connected to them. The other routers, termed *relay routers*, reside at the core of the network and only forward packets along established paths. To enhance data throughput, wireless network coding (i.e., COPE [2]) is enabled in this mesh network: routers operate in promiscuous mode and encode/decode relayed packets opportunistically. We assume that as a basic security guarantee, end-to-end encryption (e.g., SSL/TLS) has already been deployed so that attackers cannot discover the content of packets.

B. Privacy Model

We now specify the privacy goal we aim to achieve for the system defined above. For a data session that involves communication between a *sender* (i.e., the entity that originates packets) and a *receiver* (i.e., the entity for which packets are destined), three candidate privacy models given by [17] are considered:

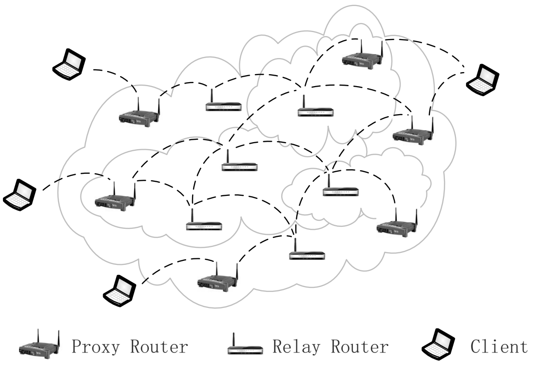


Fig. 2. An example of wireless mesh network.

- *Communication Unobservability*: an adversary cannot distinguish whether a communication exists or not.
- *Sender/Receiver Anonymity*: an adversary may observe a communication session, but cannot identify the sender/receiver of such a session.
- *Relationship Anonymity (or Unlinkability)*: an adversary may identify a sender or a receiver of some communication, but cannot determine whether they are related or not in the same session.

Note that from the above definitions, communication unobservability offers the strongest privacy guarantee, while the unlinkability offers the weakest among the three. However, in practical systems, unobservability is mainly achieved by injecting dummy/cover packets into networks, which consumes a considerable amount of network bandwidth [19], [20]. Similar performance degradation can be observed in protocols that achieve sender/receiver anonymity. Take Crowds [12] as an example, it provides sender anonymity for web transactions. However, to hide a sender's identity, other nodes in the network need to probabilistically forward the sender's packets to each other. This will incur a large delay since each packet will traverse many more extra hops before reaching the receiver. In short, providing either communication unobservability or sender/receiver anonymity requires significant network resources and may heavily degrade the performance of legitimate applications.

On the other hand, relationship anonymity can be realized with much smaller performance degradation, which is suitable for wireless networks where bandwidth resources are generally more limited as opposed to wireline networks. One successful scheme that achieves relationship anonymity is Onion Routing [9], which is inspired by Chaum's Mix [6]. In general, relationship anonymity is sufficient for most applications that require privacy preservation, since an adversary cannot deduce the sensitive information of "*who is talking to whom*", even though it can intercept traffic. Thus, to allow for practical deployment, in this paper, we choose to achieve relationship anonymity for the wireless mesh network that we consider.

C. Adversary Model

Given the relationship anonymity as the privacy property to preserve, we now define the capabilities of an adversary. The adversary we consider is *passive* in nature, i.e., it *passively* monitors network traffic, and will not drop, inject, or modify any packets. The only goal of the adversary is to deduce the information of "*who is talking to whom*" and establish the sender-receiver relationships of data sessions. To achieve this, the adversary can naively examine some identifiers (e.g., IP addresses) contained in a packet to discover the sender or receiver directly. If such identifiers are protected, the adversary can still perform traffic analysis by content-correlation, size-correlation, and time-correlation [5].

In this paper, we will classify the adversary into two categories: (1) the *external adversary*, which monitors the incoming and outgoing traffic of a target node by staying close to the target node and overhearing packets via the wireless channel; (2) the *internal adversary*, which compromises and fully controls a target node, and passively analyzes the traffic that traverses the target node. We assume that the proxy routers are trustable, in the sense that they cannot be compromised by internal adversaries.

III. Overview of General Framework

As noted in the current literature, the main task of achieving relationship anonymity is to prevent the adversary from correlating input and output packets of relay nodes. This is commonly achieved using schemes based on Chaum's mix [6], where packets are transformed before being forwarded.

In the following, we first demonstrate the *infeasibility* of Chaum's mix-based schemes in wireless network coding, and then present the design rationale of our proposed scheme. For clarity of explanation, we adopt the paradigm of *Onion Routing* [9], a classic mix-based scheme that is the core of many prior anonymous protocols [10], [11], [15]. However, we emphasize that other mix-based schemes can also be used as the building block of our proposed scheme in a similar fashion.

A. Infeasibility of Onion Routing in Network Coding

Onion Routing [9] is an anonymous routing protocol that can achieve relationship anonymity in traditional networks without network coding. A typical Onion Routing system consists of inter-connected routers called *onion routers*. Each router i is loaded with a pair of public/private keys (uk_i, rk_i), and the global knowledge of the network topology. In the following, we briefly describe how Onion Routing works when no network coding is used, using the simple cross topology shown in Fig. 1(a).

Suppose that two end users U_1 and U_2, who are respectively connected to routers S_1 and D_1, want to set up a session. In Onion Routing, U_1 first sends a connection request to S_1. On receiving this request, S_1 determines a path to router D_1 (in this case, the path is simply $S_1 \rightarrow C \rightarrow D_1$). Then S_1 randomly selects two session keys sk_{C1} and sk_{D1} for C and D_1 respectively, and constructs a layered data structure called an *onion* as $\{\{sk_{D1}, U_2\}_{uk_{D1}}, sk_{C1}, D_1\}_{uk_C}$, where uk_C and uk_{D1} are the public keys of C and D_1, respectively, and $\{\cdot\}_k$ denotes the encryption using public key k. Then S_1 sends this onion to C, which uses its private key rk_C to decrypt the onion. After decryption, C will obtain its session key sk_{C1}, the next-hop router D_1, and the embedded onion $\{sk_{D1}, U_2\}_{uk_{D1}}$. This embedded onion is then forwarded to D_1, which decrypts it using its private key to get the session key sk_{D1}. In addition, D_1 will find that it is the last hop of the route, as the next hop is the end user U_2 connected to D_1. Then D_1 forwards the connection request to U_2, and a data session is established. After the route establishment, data is transmitted using symmetric-key encryptions. Specifically, using the session keys previously assigned, S_1 applies symmetric-key encryption to each message M originated from U_1 and constructs $\{\{M\}_{sk_{D1}}\}_{sk_{C1}}$. Then C removes the outermost layer using sk_{C1} to get $\{M\}_{sk_{D1}}$, and finally D_1 removes the innermost layer using sk_{D1} to recover message M.

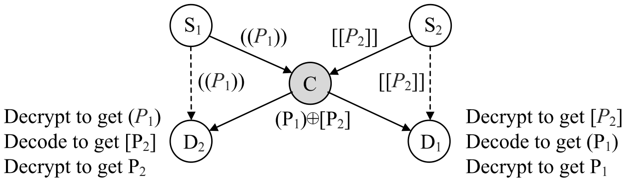


Fig. 3. An illustration of how relay nodes cooperate to make Onion Routing and network coding compatible. $(\cdot)$ and $[\cdot]$ denote the symmetric-key encryptions performed by S_1 and S_2 on their data packets, respectively.

Similarly, we can apply Onion Routing for another session that uses path $S_2 \rightarrow C \rightarrow D_2$. We can assign C and D_2 session keys sk_{C2} and sk_{D2} for this session, respectively.

Suppose that network coding is enabled. We now show *how Onion Routing fails*. First, D_1 and D_2 can overhear the packets $\{\{P_1\}_{sk_{D1}}\}_{sk_{C1}}$ and $\{\{P_2\}_{sk_{D2}}\}_{sk_{C2}}$ from S_1 and S_2, respectively, and both packets will be received by C as well. Then, C will perform decryption on these two packets and get $\{P_1\}_{sk_{D1}}$ and $\{P_2\}_{sk_{D2}}$. By network coding, C would broadcast $\{P_1\}_{sk_{D1}} \oplus \{P_2\}_{sk_{D2}}$. However, *neither* D_1 *nor* D_2 *can decode the packets*, as they only overhear the packets encrypted with session keys sk_{C1} and sk_{C2} possessed by C, respectively.

Finally, it is important to note that this simple example provides an illustrative insight for larger topologies. Suppose that an adversary can eavesdrop traffic that traverses node C. When Onion Routing is used, the adversary can only tell the previous hops (i.e., S_1 and S_2) and next hops (i.e., D_1 and D_2) of node C, but cannot determine the nodes that are further upstream or downstream. Such a privacy guarantee cannot be directly achieved with simple end-to-end encryption.

B. Design Rationale of ANOC

Cooperative networking is a relatively new design policy which encourages multiple nodes to cooperate to finish a common communication goal, and is successfully applied to wireless ad hoc networks [21], [22] and content distribution networks [23]. We observe that the idea of cooperative networking can also be used here to resolve the conflicts between Onion Routing and network coding. Taking the cross topology for example again, the following two-step cooperation (as illustrated in Fig. 3) can help Onion Routing adapt to network coding.

(1) *session-key sharing*: C shares its session key sk_{C1} and sk_{C2} with D_2 and D_1, respectively;

(2) *auxiliary decrypting*: D_2 decrypts the overheard packet $((P_1))$ using sk_{C1} to obtain (P_1), and D_1 decrypts the overheard packet $[[P_2]]$ using sk_{C2} to obtain $[P_2]$.

After the above cooperation, C can broadcast the coded packet $(P_1) \oplus [P_2]$. Then, D_1 can decode this packet to get (P_1), and decrypt (P_1) to get P_1; similarly, D_2 can obtain P_2. In this way, the conflict between network coding and Onion Routing is resolved.

Now, we consider two different approaches to achieve session-key sharing between the coding node (C in the example) and the decoding nodes (D_1 and D_2 in the example). The first but naive approach is to simply let each router share its private key with all its one-hop neighbors (e.g., C shares rk_C with D_1 and D_2), so that when an anonymous session passing through the router is established, each of its neighbors can also obtain the session key. This approach can be carried out during the establishment phase of an anonymous session, and hence will not incur any online overhead. However, the sharing of private keys would severely undermine the security of system.

For the second approach which we are going to adopt, the coding node shares its session keys (instead of private keys) with its one-hop neighbors in an on-demand fashion. Specifically, when there are coding opportunities, the router that performs coding should securely broadcast its session keys of the corresponding sessions to its neighbors. One critical point is that the share of session keys is only limited to neighboring nodes. Nodes that are further upstream or downstream cannot see the session keys; otherwise the user privacy cannot be properly preserved. Another critical point is that the key sharing procedure is only triggered when there are opportunities for network coding, so that session keys will not be shared unnecessarily.

IV. ANOC: The Details

We propose ANOC, the anonymous network-coding-based communication for wireless mesh networks. ANOC is built upon the traditional Onion Routing protocol, and introduces efficient cooperation (i.e., session-key sharing and auxiliary decrypting) among relay nodes to resolve the conflict between Onion Routing and network coding. The technical challenges include: (i) how to trigger the session-key sharing in an on-demand fashion, and (ii) how to efficiently and securely share session keys with neighbors without leaking any information to adversaries. In the following, we show how ANOC addresses these two challenges.

A. System Setup

First, each of the routers (including proxy routers and relay routers as shown in Fig. 2) is assigned a unique router identifier and preloaded with a pair of public/private keys. In particular, each proxy router knows about the network topology and the public keys of all other routers in the network; each relay router only knows about its neighboring routers and their public keys. Also, each router maintains a sufficiently large buffer for bathing and reordering packets, such that the time-correlation of its incoming and outgoing packets can be eliminated (see [9] for details).

In the bootstrap stage of ANOC, each router performs operations offline to establish the secure broadcast key and local neighboring table. These operations are explained below.

1) Secure Broadcast Key: Each router R randomly selects its *broadcast key*, which will be later used for link-layer encryptions of (i) packet headers and (ii) distribution of session-keys. For each of the neighboring routers, R encrypts its broadcast key using the public key of the neighbor and unicasts the ciphertext to that neighboring router.

2) Local Neighboring Table: Each router R maintains a *local neighboring table* that records the neighbors of each of R's neighbors. The table will be used to determine whether there are coding opportunities (details will be presented in Section IV-D). For instance, for node C in Fig. 1(c), its local neighboring table will specify that node S_1 has neighbors S_2 and S_4. The table can be easily constructed by having each router broadcast the list of its one-hop neighbors.

B. Packet Format

ANOC assumes that wireless network coding (i.e., COPE [2]) is enabled, and the packet header of COPE is placed right after the MAC header. In addition, we add a new routing header to enable anonymous routing. Fig. 4(a) illustrates the layout of the COPE header and the routing header in our protocol. The routing header consists of two fields: COMMAND and CIR_ID. The COMMAND field describes the type of a packet. In ANOC, there are four types of packets:

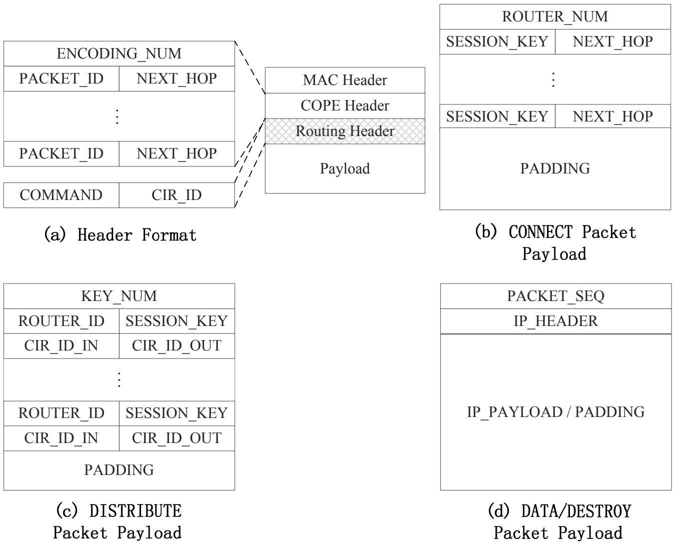
- CONNECT (Section IV-C): for route establishment,
- DISTRIBUTE (Section IV-D): for session-key distribution,
- DATA (Section IV-E): for information delivery, and
- DESTROY (Section IV-F): for tearing down an existing session.

The CIR_ID field carries the *circuit identifier* which enables multiple sessions to be multiplexed into a single physical channel. We explain its use in Section IV-C.

Figs. 4(b)-(d) illustrate the four types of packets (i.e., CONNECT, DISTRIBUTE, DATA, and DESTROY), each of which is attached with different payload fields that are encrypted with different types of keys. We will explain each type of packets and how each is encrypted in the following subsections. In particular, we encrypt the COPE header and the routing header with the broadcast keys that have been established during the bootstrap phase (see Section IV-A). Therefore, the sensitive information such as the packet type will not be disclosed to external adversaries. Furthermore, we fix each packet to have the same size using padding so as to prevent an adversary from inferring a session through size correlation [9].

C. Session Setup

To setup a new session, the proxy router first selects the path of routers in the network toward the destination. It then generates a CONNECT packet, which specifies the selected



(a) Header Format

(b) CONNECT Packet Payload

(c) DISTRIBUTE Packet Payload

(d) DATA/DESTROY Packet Payload

Fig. 4. The header and payload format of ANOC.

routers and the corresponding session keys for each of the routers. Each router and its corresponding session key will be encrypted with the public key of the router, such that the CONNECT packet forms an onion structure (refer to Section III-A).

In addition, when initiating a session, the proxy router randomly chooses a locally unique number (i.e., circuit identifier) to identify the session. This number is placed in the CIR_ID field of the CONNECT packet. When a downstream router receives the CONNECT packet, it will record the circuit identifier in the CIR_ID field, and choose a new circuit identifier and replace the CIR_ID field with it. In this way, each router maintains a circuit-identifier mapping for the session. This mapping will later enable DATA packets to be routed along the path specified in the CONNECT packet.

D. Session-Key Sharing: The First Step of Cooperation

In ANOC, session keys are shared in an on-demand manner based on coding opportunities. To discover a coding opportunities, we adopt the flow-based[1] coding conditions given in [24]. Using these conditions, we can determine whether packets from $n \geq 2$ flows can be coded together. For more details, please refer to our technical report [16].

After discovering the coding opportunities, the router can start sharing its session keys. Suppose that there are n coding flows $F_1, \ldots, F_n$ intersecting at router C, then C should distribute its session keys associated with $F_1, \ldots, F_n$ to all its one-hop neighbors. This is achieved by broadcasting a DISTRIBUTE packet encrypted with C's broadcast key, which is established in the system setup stage (refer to Section IV-A). In addition to the session keys, the DISTRIBUTE packet also contains the incoming and outgoing circuit identifiers, in order that all neighbors of the coding node can properly process overheard packets (see Section IV-E). As shown in Fig. 4(c), a DISTRIBUTE packet contains the tuples (Router_ID, Session_Key, CIR_ID_IN, CIR_ID_OUT), where Router_ID is the identifier of the upstream router of C in this route, Session_key is the session key for C in this route, and CIR_ID_IN and CIR_ID_OUT are the incoming and outgoing circuit identifier for the session, respectively. For instance, let us consider the coding scenario in Fig. 3. Let the mapping of circuit identifiers in node C be $001 \rightharpoonup 255$ and $102 \rightharpoonup 123$ for sessions $S_1 \rightarrow C \rightarrow D_1$ and $S_2 \rightarrow C \rightarrow D_2$, respectively, and let C hold the session keys sk_{C1} and sk_{C2} for these two sessions, respectively. Then the two tuples contained in the DISTRIBUTE packet would be (S_1, sk_{C1}, 001, 255) and (S_2, sk_{C2}, 102, 123).

E. Auxiliary Decrypting: The Second Step of Cooperation

In ANOC, we implement auxiliary decrypting (see Section III-B) via a separate module named *overhearing module*.This module consists of a key table, an overheard packet pool, and a decrypting unit. The key table stores the tuples (Router_ID, Session_Key, CIR_ID_IN, CIR_ID_OUT) of all DISTRIBUTE packets received from neighbors (see Section IV-D). When a router sends a packet, or overhears a packet that is destined to a different MAC address rather than itself, it will find the router identifier of the sender (by mapping to the source MAC address) and the circuit identifier contained in the packet header. It then looks up in the key table for the tuple that has the mapping indexed by (Router_ID, CIR_ID_IN). If no tuple is found, then the overheard packet will be discarded; otherwise, if the tuple is found, then the router will decrypt the overheard packet using the session key in the tuple, and replace the CIR_ID field of the overheard packet with CIR_ID_OUT in the tuple. The resulting packet will be stored in the overheard packet pool. When later the router receives a coded packet, it will look up in the overheard packet pool for packets that are necessary for decoding.

F. Session Teardown

When an initiator needs to tear down one of its sessions, it will send a DESTROY packet along the route. Upon receiving the DESTROY packet, each router en route deletes all the information for the session. The neighboring routers will also expire the session keys for the session (received through DISTRIBUTE packets) after a pre-specified timeout period.

V. Privacy Enhancement on ANOC

In this section, we analytically show how ANOC enhances privacy over a wireless mesh network that enables network coding. We consider the external adversary and the internal adversary in our adversary model defined in Section II-C.

A. The External Adversary

We argue that ANOC can achieve relationship anonymity against the external adversary, which monitors packets via overhearing the wireless channel and attempts to perform traffic analysis via correlations of content, size, and time. We give our justifications as follows.
(a) *Content-correlation*. In ANOC, any CONNECT, DATA or DESTROY packet will undergo encryption or decryption when passing through each router. Thus, correlation based on

[1] Here, a flow is equivalent to a session.

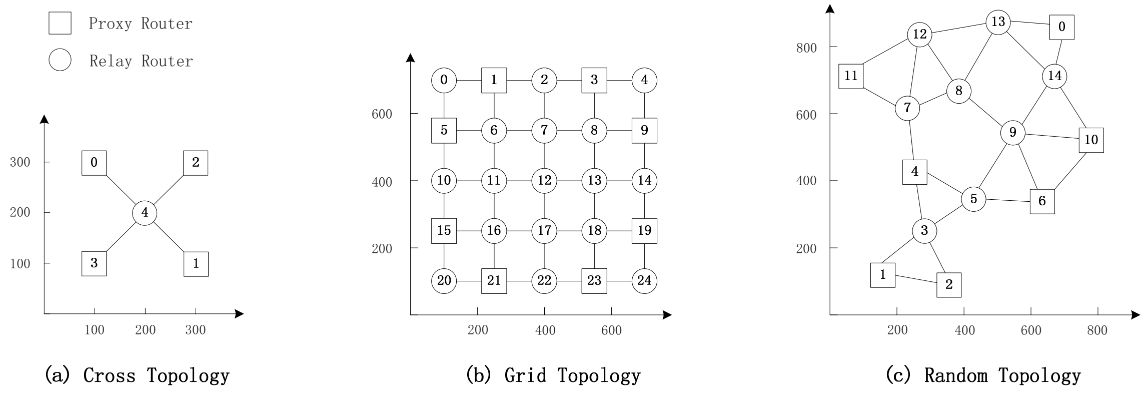


Fig. 5. Three topologies used in our experiments.

content will be impossible. As for DISTRIBUTE packets, they are encrypted (using the secure broadcast key) and broadcasted without revealing the receiver identities or routing information. (b) *Size-correlation*. In ANOC, each packet is padded into the same size. Thus, it is impossible to perform traceback by correlating packet sizes.
(c) *Time-correlation*. With batching and reordering operations performed at each router, a packet cannot be associated with others by examining the sending and receiving time.

B. The Internal Adversary

In Onion Routing, the whole path of a specific route is known by the involved proxy routers, while the relay routers along the route can only identify their previous and next hops. This means that if one relay router is compromised, then it cannot expose the sender-receiver relationship of the whole route. Clearly, this argument holds in ANOC as well when there are no coding opportunities (as in Onion Routing). On the other hand, when a coding node distributes its session keys using secure broadcast, its one-hop neighbors will inevitably obtain more information. This can allow the internal adversary to discover more hops in addition to its previous and next hops for a given anonymous session.

In our technical report [16], we show how the internal adversary can leverage the session-key sharing in our ANOC protocol to discover more hops in a session. We also give analytical results to demonstrate the number of additional hops discovered by the internal adversary is *rather limited.* Specifically, we show that the expected number of additional upstream nodes that can be identified by an internal adversary A is $q(1-q^d)/(1-q)$, where q is the probability that a given node is a neighbor of A, and it has a coding opportunity. For detailed discussion, please refer to [16].

VI. Experimental Results

We now evaluate ANOC using realistic wireless settings. Our evaluation is based on *nsclick* [18], which embeds Click Modular Router [25] into the ns2 simulator [26].

We design two Click modules to reflect our system model: the *proxy router module*, which defines a proxy router for initiating new sessions and selecting routes, and the *relay router module*, which defines a relay router for forwarding data packets and performing network coding.

For comparisons with ANOC, we also implement the following two routing protocols:

- *COPE*, the routing protocol with network coding but no anonymity protection,
- *Onion*, the Onion Routing protocol with anonymity protection but no network coding.

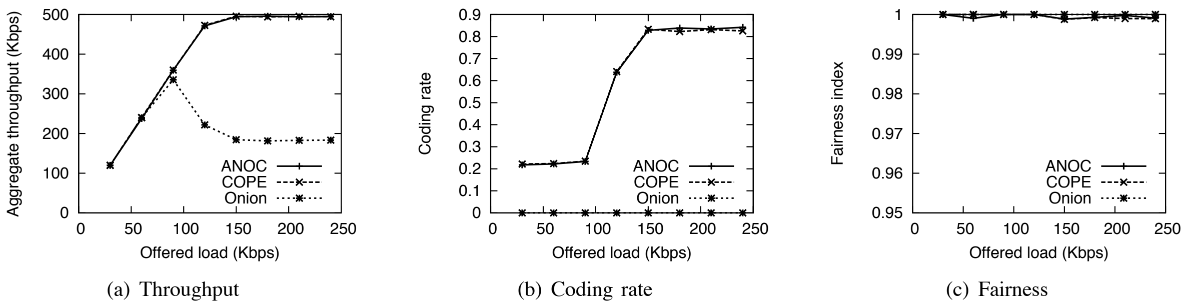
Our experiments focus on the following four metrics:

- *throughput*, the aggregate throughput of all sessions in the network,
- *coding rate*, the ratio of the number of encoded packets to the total number of forwarded packets,
- *fairness*, the measure of how peer flows get equal throughput based on Jain's fairness index [27] $(\sum x_i)^2/(N\sum x_i^2)$ (where x_i denotes the throughput of the ith flow and N is the number of all flows), and
- *symmetric-key encryption/decryption*, the computational cost of packet processing when anonymity protection is used.

We carry out experiments using three representative topologies given in Fig. 5: (a) the cross topology, which is relatively simple and serves as the baseline setting, (b) the grid topology, representing a relatively complex but regular setting, and (c) the random topology, a more realistic setting to test the applicability of our scheme.

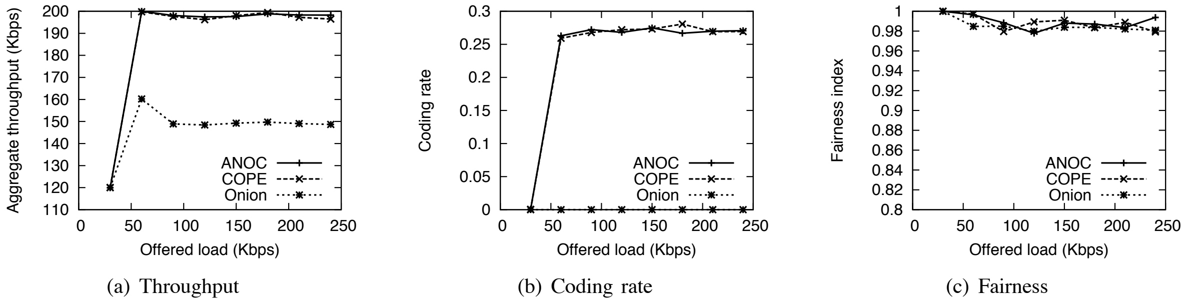
In the following, we will consider the throughput, coding opportunity, and fairness for each of the topologies with all three routing protocols. Actually we also measure the encryption/decryption performance specifically for Onion and ANOC, and we put the results in our technical report [16], due to limited space.

Experiment 1. Cross Topology: We first revisit the simple cross topology (see Fig. 5(a)), in which we create four sessions: 0 → 1, 1 → 0, 2 → 3, and 3 → 2, all of which are relayed by router 4. We assume that these four flows have the same offered load. Fig. 6 shows the performance of different routing protocols. From Fig. 6(a), we observe that when the offered load increases, the aggregate throughput achieved by ANOC also increases and is fairly close to that of COPE, while the throughput of Onion drops. The reason is that when the offered load increases, there is a higher coding



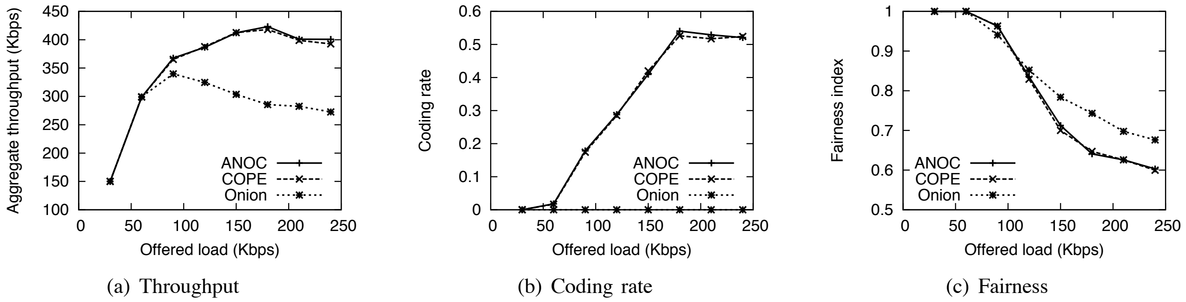
(a) Throughput (b) Coding rate (c) Fairness

Fig. 6. Experiment 1: Throughput, coding rate, and fairness of three protocols in the cross topology.



(a) Throughput (b) Coding rate (c) Fairness

Fig. 7. Experiment 2: Throughput, coding rate, and fairness of three protocols for the grid topology.



(a) Throughput (b) Coding rate (c) Fairness

Fig. 8. Experiment 3: Throughput, coding rate, and fairness of three protocols for the random topology.

opportunity for ANOC and COPE, as confirmed by Fig. 6(b), while ANOC suffers many packet collisions under the high offered traffic load. Also, Fig. 6(c) shows that both ANOC and COPE achieve a high fairness index.

Experiment 2. Grid Topology: We proceed to study the complex but regular grid topology given by Fig. 5(b), in which there are four sessions: $5 \rightarrow 9$, $15 \rightarrow 19$, $1 \rightarrow 21$, and $3 \rightarrow 23$. We deploy the 25 nodes in a way that the radio transmission range of each node covers all neighboring nodes along its surrounding square (i.e., a node can have at most eight neighboring nodes). This topology differs from the previous cross topology in that each routing path consists of more than two hops. Fig. 7 shows the obtained results. Similar to Experiment 1, ANOC and COPE have similar performance and they both outperform Onion when the offered load is high.

Experiment 3. Random Topology: We then consider a 15-node random topology where nodes are randomly placed over a plane, as shown in Fig. 5(c). We pick five sessions for our evaluation: $0 \rightarrow 10$, $10 \rightarrow 0$, $1 \rightarrow 6$, $4 \rightarrow 2$, and $6 \rightarrow 11$. Fig. 8 shows the results we get, which are mostly consistent with the results in previous experiments. The only difference we want to highlight is that the fairness indices for ANOC and COPE decrease when the offered load increases. The primary reason is due to the asymmetric property of this random topology, such that the sessions $0 \rightarrow 10$, $10 \rightarrow 0$, $1 \rightarrow 6$, and $4 \rightarrow 2$ receive a higher coding opportunity when the offered load increases, while the session $6 \rightarrow 11$ does not. However, this decrease in fairness is independent of the use of anonymous routing.

VII. Conclusions

In this paper, we point out an important problem that when a wireless network enables network coding, previously functioning privacy-preserving schemes may no longer perform correctly. To this end, we propose ANOC, a novel anonymous communication protocol which can function seamlessly with wireless network coding. Analytical and experimental results imply that our ANOC not only keeps the advantage of network coding in the effective use of wireless network capacity, but also provides privacy for wireless users at the same time.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[2] S. Katti, H. Rahul, D. Katabi, W. Hu, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 497–510, Jun. 2008.

[3] I. F. Akyildiz and X. Wang, "Wireless mesh networks: a survey," *Computer Networks*, vol. 47, no. 4, pp. 445–487, 2005.

[4] J. Le, J. C. S. Lui, and D. M. Chiu, "How many packets can we encode: An analysis of practical wireless network coding," in *Proc. IEEE INFOCOM'08*, Apr. 2008.

[5] A. Back, U. Möller, and A. Stiglic, "Traffic analysis attacks and trade-offs in anonymity providing systems," in *Proc. International Workshop on Information Hiding*, Apr. 2001.

[6] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981.

[7] G. Danezis, R. Dingledine, and N. Mathewson, "Mixminion: Design of a Type III anonymous remailer protocol," in *Proc. IEEE Symposium on Security and Privacy*, May 2003.

[8] M. Rennhard and B. Plattner, "Introducing MorphMix: Peer-to-Peer based anonymous internet usage with collusion detection," in *Proc. ACM Workshop on Privacy in the Electronic Society*, 2002.

[9] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, "Anonymous connections and onion routing," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 4, pp. 482–494, May 1998.

[10] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proc. 13th USENIX Security Symposium*, 2004.

[11] S. Katti, J. Cohen, and D. Katabi, "Information slicing: Anonymity using unreliable overlays," in *Proc. NSDI*, 2007.

[12] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM Trans. Information and System Security*, vol. 1, no. 1, pp. 66–92, Nov. 1998.

[13] J. Kong and X. Hong, "ANODR: Anonymous on demand routing with untraceable routes for mobile ad-hoc networks," in *Proc. ACM MobiHoc*, Jun. 2003.

[14] M. Blaze, J. Ioannidis, A. D. Keromytis, T. Malkin, and A. Rubin, "Anonymity in wireless broadcast networks," *International Journal of Network Security*, vol. 8, no. 1, pp. 37–51, Jan. 2009.

[15] X. Wu and N. Li, "Achieving privacy in mesh networks," in *Proc. ACM Workshop on Security of Ad Hoc and Sensor Networks*, Oct. 2006.

[16] P. Zhang, Y. Jiang, C. Lin, P. P. C. Lee, and J. C. S. Lui, "Anoc: Anonymous network-coding-based communication with efficient cooperation," Tech. Rep., http://qos.cs.tsinghua.edu.cn/ pzhang/jsac-tr.pdf.

[17] A. Pfitzmann and M. Hansen, "Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management – a consolidated proposal for terminology," http://dud.inf.tu-dresden.de/Anon_Terminology.shtml, Feb. 2008, v0.31.

[18] M. Neufeld, G. Schelle, and D. Grunwald, "Nsclick user manual," University of Colorado, Boulder, CO 80309, Tech. Rep., Aug. 2003.

[19] Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, and G. Cao, "Towards event source unobservability with minimum network traffic in sensor networks," in *Proc. ACM WiSec*, Mar. 2008.

[20] M. Shao, Y. Yang, S. Zhu, and G. Cao, "Towards statistically strong source anonymity for sensor networks," in *Proc. IEEE INFOCOM*, Apr. 2008.

[21] A. Sendonaris, E. Erkip, and R. Aazhang, "User cooperation diversity. Part I. System description," *IEEE Trans. Commun.*, vol. 51, no. 11, pp. 1927–1938, 2003.

[22] ——, "User cooperation diversity. Part II. Implementation aspects and performance analysis," *IEEE Trans. Commun.*, vol. 51, no. 11, pp. 1939–1948, 2003.

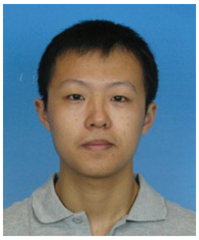
[23] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networkin," in *Proc. 12th international workshop on Network and operating systems support for digital audio and video*, 2002.

[24] J. Le, J. C. S. Lui, and D. M. Chiu, "DCAR: Distributed coding-aware routing for wireless networks," *IEEE Trans. Mobile Computing*, vol. 9, no. 4, Apr. 2010.

[25] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Trans. Computer Systems*, Aug. 2000.

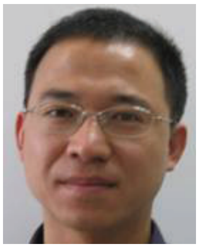
[26] "The network simulator - ns-2," http://www.isi.edu/nsnam/ns/.

[27] R. Jain, *The Art of Computer Systems Performance Analysis*. Wiley-Interscience, 1991.

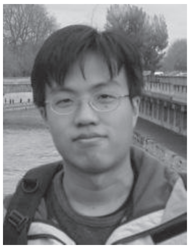


Peng Zhang received the B.Eng. degree in Computer Science from Beijing University of Posts and Telecommunications in 2008. He is now a Ph.D. student in the Department of Computer Science and Technology at Tsinghua University. He was a visiting student in the Department of Computer Sicence and Engineering at the Chinese University of Hong Kong between 2009 and 2010. His research interests include network coding and network security.

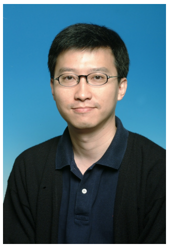
Chuang Lin received the Ph.D. degree in Computer Science from Tsinghua University in 1994. He is now a professor of the Department of Computer Science and Technology, Tsinghua University, China. He is a Honorary Visiting Professor, University of Bradford, UK. His current research interests include computer networks, performance evaluation, network security analysis, and Petri net theory and its applications. He has published more than 300 papers in research journals and IEEE conferences, and four monographs in these areas.



Yixin Jiang received the Ph.D. degree in Computer Science from Tsinghua University in 2006. He is now an associate professor at Tsinghua University. In 2005, he was a Visiting Scholar with the Department of Computer Science, Hong Kong Baptist University. From 2007 to 2009, he was a Post Doctorial Fellow with University of Waterloo. His research interests include wireless network security, trusted computing and network coding.



Patrick P.C. Lee received the B.Eng. degree (first-class honors) in Information Engineering from the Chinese University of Hong Kong in 2001, the M.Phil. degree in Computer Science and Engineering from the Chinese University of Hong Kong in 2003, and the Ph.D. degree in Computer Science from Columbia University in 2008. He is now an assistant professor of the Department of Computer Science and Engineering at the Chinese University of Hong Kong. His research interests are in network robustness and security.



John C.S. Lui received his Ph.D. in Computer Science from UCLA. He is currently a professor of the CS&E Department at The Chinese University of Hong Kong. His current research interests are in data networks, system and applied security, multimedia systems, network sciences and cloud computing. He is an associate editor in the Performance Evaluation Journal, IEEE-TC, IEEE-TPDS and IEEE/ACM Transactions on Networking. John is a Fellow in ACM and IEEE, and president of ACM SIGMETRICS.