Achieving Privacy in Mesh Networks

Xiaoxin Wu
Intel China Research Center Ltd
Beijing, China
xiaoxin.wu@intel.com

Ninghui Li
Department of Computer Science
Purdue University
West Lafayette, IN 47907-2086, USA
ninghui@cs.purdue.edu

ABSTRACT

Mesh network is vulnerable to privacy attacks because of the open medium property of wireless channel, the fixed topology, and the limited network size. Traditional anonymous routing algorithm cannot be directly applied to Mesh network, because they do not defend global attackers. In this paper we design private routing algorithm that used "Onion", i.e., layered encryption, to hide routing information. In addition, we explore special ring topology that fits the investigated network scenario, to preserve a certain level of privacy against a global adversary.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing protocols*

General Terms

Security, Design

Keywords

Mesh networks, Privacy

1. INTRODUCTION

Mesh network [1, 2, 3, 4, 5] has been proposed to be the solution for the last mile of network communications because it is able to provide low-cost, high-speed network services to the end users. A Mesh network can be deployed in an environment where there is no existing wired network to the end users, or the capacity for the existing network is insufficient such that the Mesh network provides a supplementary service. For example, a Mesh network may be constructed in a rural community so that the community can share one satellite Internet connection. In such a network, each Mesh node is a household with wireless equipments, and there is a Gateway router that is connected to the Internet. Mesh nodes can communicate with each other, and access the Internet through the gateway router. The wireless connection between Mesh nodes and its Gateway router can be single hop or multi-hop. When multi-hop connection is needed, a Mesh node connects to the Gateway router with the aid of other Mesh nodes that act as intermediate forwarders. An example of such a Mesh network is illustrated in Fig. 1 (a).

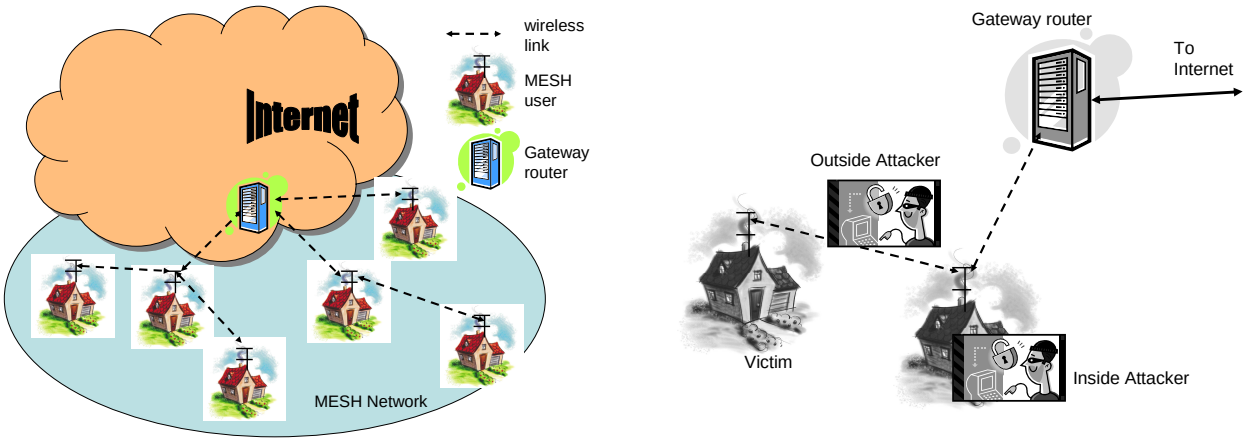
In such a Mesh network, as all traffic goes through the Gateway router, the Gateway router may act as a centralized control, as an access point in WLAN [6], or a base station in the cellular network. A network with centralized control has the advantages over a self-organized network because it can have better resource allocation and routing optimization. It also makes security and privacy issues less challenging. In the Mesh network, a Mesh node then cannot initiate a session at its will. Instead, it has to send the Gateway router an access request. The Gateway router grants an access and sends an assigned route, through which the Mesh node connects to the Internet.

The behavior of a Mesh node can be easily monitored or traced by adversaries due to the use of wireless channel, multi-hop connection through third parties, and converged traffic pattern going through the Gateway router. In this paper we consider a global adversary model that is made up by colluded inside and outside attackers, as illustrated in Fig. 1 (b). Under such an adversary model, the primary privacy objective we want to achieve is hiding an active node that connects to the Gateway router among a group of Mesh nodes. In other words, the active Mesh node has to be anonymous. Such a protection is important when on the Internet side, traditional anonymous routing approaches are not implemented, or may be compromised by strong attackers.

Traditional private communication approaches designed for wired networks apply either cryptography [7, 8, 9] or redundancy to achieve communication end privacy [11, 12, 13]. Cryptographic approaches cannot be adopted directly to achieve our privacy goal in the Mesh network because they are not efficient under a global attack. Redundancy approaches, e.g., a broadcast at the Gateway router, may keep a receiving node anonymous. Yet as most communications in the Mesh network are bi-directional, a global attacker can still discover the node when it sends a message to the Gateway router. Adding background noise for preserving sender privacy is expensive especially in wireless networks.

To solve the above problem, we design a novel communication protocol, called Onion Ring, to defend against a global, aggressive attacker and to protect node privacy by using both cryptography and redundancy. Such an approach explores the special topology of the Mesh network, i.e., all

SASN'06, October 30, 2006, Alexandria, Virginia, USA.



(a) Mesh network

(b) Privacy weakness

Figure 1: Mesh network and privacy weakness

the traffic goes through the Gateway router. The extra computing load caused by using cryptography is not a concern for Mesh networks because Mesh nodes are desktop or laptop computers with sufficient power supply and computing capability. On the other hand, the network performance degradation caused by redundant transmissions can be mitigated because the proposed approach enables centralized control, which facilitates network optimization such as global scheduling.

The paper is organized as follows. In Section 2 we list network assumptions, security assumptions, attacker models, and privacy goals. In Section 3 we review Onion routing. In Section 4 we present redundant Onion routing in the Mesh network. In Section 5 we present Onion ring that can defend against a global adversary. In Section 6 we address the intruder identification scheme that discovers the compromised Mesh nodes. In Sectionsix we present related works. In Section 8 we conclude and discuss future research directions.

2. ASSUMPTIONS, ATTACK MODELS, AND PRIVACY GOAL

2.1 Network and Security Assumptions:

Network Assumptions We assume that the wireless channel is symmetric, i.e., if A can send data to B through a multi-hop wireless path, B can send data to A using the reverse path. We assume that error control is used at the link layer, therefore an erroneous packet is not caused by wireless transmission. We assume that the Gateway router knows the network topology. This information can be obtained because when any new Mesh node joins the network, an routing update is required. It discovers its neighbors first. Then it uses RIP [14] or OPSF [15] to find a route to the Gateway router and update the network topology.

Security Assumptions We assume that a PKI local to the Mesh network is in place when the Mesh network is set up. For example, the Gateway router possesses a pair of public/private keys and the public key is configured into each Mesh node. We assume that the Gateway router is difficult to be compromised and is thus trusted. Each Mesh node also possesses a pair of public/private keys and the public key is certified by the Gateway router. In other words, the Gateway router acts as a Certification Authority (CA) local to the Mesh network. Using these public keys, a symmetric key is established between the Gateway router and each Mesh node.

We assume that a Mesh node also establishes a symmetric key with any of its one-hop neighboring Mesh node. The communication between any two neighboring nodes then is confidential. However, plain text has to be used in the packet header to indicate the identity of the sender and the receiver [1].

2.2 Attack Models and Privacy Goal

Refer to Fig. 1 (b), we list the capabilities for the following two different kinds of attackers.

- *Inside Attacker (Attacker 1)*: An inside attacker is a Mesh node that is included (probably as a forwarder) in a Mesh connection. Therefore, it knows who is its previous hop and who is its next hop. It also knows the type of the packets going through itself, e.g, whether the packet is a data packet or a control packet. In particular, when non-private routing protocol is used, the insider attacker is able to know the communication ends for the Mesh connection.

- *Outside attacker (Attacker 2)*: An outside attacker is not a registered Mesh user. It monitors a Mesh node

[1]The transmissions on link layer can be anonymous, i.e., without the header of sender and receiver information. However, this results in high computing overhead because every neighbor of a sender has to decrypt the data using the keys that it shared with all of its neighbors, to find out whether the packet is for itself.

or the Gateway router by staying closely to its target. It is able to obtain the link layer communication ends, i.e., the identities of the sender and the receiver at any hop. However, an outside attacker does not know the packet type. An attacker that monitors the behavior of the Gateway router is able to tell which Internet addresses the Gateway router is communicating with.

We assume the existence of a small number of inside attackers that collude with a global outside attacker [2]. Such an adversary can monitor the behavior of each Mesh node, such as a message transmitting or receiving, and for some circumstances, it can obtain extra information such as the type of a message. We assume that the global adversary may conduct aggressive behavior to break Mesh node privacy. For example, an inside attacker can inject traffic in the route or manipulate messages. The outside attacker can jam the Mesh nodes one by one, to find the nodes that are communicating with the Internet.

Our privacy goal is to prevent the instant behavior of any Mesh user from being observed by adversaries. The instant behavior means a specific action, i.e., a Mesh user is visiting an Internet address. We provide a certain level of anonymity, i.e., to hide the active Mesh node among a group of nodes. This means that even, if the attacker can discover which website the Gateway router is connected to, it cannot determine the individual Mesh node that is active. Adversaries then cannot build user profiles such as what time a user normally accesses the Internet and what Internet address this user visits. The size of the group can be adjusted to achieve different level of anonymity. There is a tradeoff between the anonymity and the computing/communication overhead. Achieving a higher level of anonymity will result in higher overhead cost. We avoid introducing constant background noise traffic in a Mesh network, as this can be very expensive and wastes much bandwidth and energy. As a result, the adversary would know that one of the Mesh nodes is active.

3. ONION ROUTING OVERVIEW

In this section we first review the Onion routing approach that is designed for anonymous communication in wired networks. We then discuss in which ways Mesh networks differ from wired networks and why the Onion routing approach does not solve the privacy problem in Mesh networks.

In the rest of the paper, we denote $E_k()$ as using key k to encrypt, $E_k^{-1}()$ as using key k to decrypt. k_{pi} stands for i's public key. k_{ij} stands for a symmetric key between i and j. For the sake of presentation, we call a packet that is not in plain text, i.e., a packet has been encrypted or decrypted for one or more than one time, an "Onion". k_i stands for Node i's session key for encrypting/decrypting "Onion"s.

3.1 Onion Routing and Padding Techniques

The Onion routing [8] achieves communication privacy by making communication ends un-linkable. An Onion routing network consists of a number of inter-connected Onion routers (ORs); each OR has a pair of public/private keys. Each OR knows the topology of the Onion network as well as the public keys of other ORs. An end user that requires an anonymous communication will send a request to an OR that it trusts; this OR is known as the Onion Proxy (OP) for the user. The communication between an end user and its OP is protected from the adversaries. The OP determines a route that consists of a series of ORs and constructs an "Onion" using the public keys of the routers en route.

The "Onion" is constructed in a way such that the most inner part is the message for the intended destination. The message is *wrapped*, i.e., encrypted using the public keys of the ORs in the route, in the same order as the ORs appear in the route. Once an OR receives the Onioned message, it uses its private key to *peel*, i.e., decrypt, the "Onion", to obtain the information such as the next hop and the session key. It then forwards the rest of the "Onion" to the next hop. This process is repeated until the "Onion" reaches the last OR, which peels the last layer of the "Onion" and obtain the *exit* information, i.e., the destination.

For example, if the private route is $R_1 \rightarrow R_2 ... \rightarrow R_n$, where R_i is the ith OR, and the last router R_n will connect to the exit funnel of the ORs, which will further communicate with the address requested by the session initiator; the message flow and the "Onion"s received at each router in the route are as follows:

$$E_{k_{pR_1}}(R_2, k_1, E_{k_{pR_2}}(...E_{k_{pR_n}}(k_n, exit)...)) \quad (1)$$
$$\rightarrow E_{k_{pR_2}}(...E_{k_{pR_n}}(k_n, exit)...) ... \rightarrow E_{k_{pR_n}}(k_n, exit).$$

k_{pR_i} and k_i are the public key and assigned session key for the ith router. After the route is built up, session keys are used for constructing "Onion"s, and anonymous circuit ID (ACI) is used for routing. For the reverse path, data packet was encrypted by the session keys. The OP receives the "Onion" in the reverse path and peels it using the session keys it assigned to the ORs, and sends the raw data to the end user.

For an Onion route, only the proxy knows the first and the last router. Any OR in the route only knows its previous hop and next hop. For both outside attackers and inside attackers (i.e., compromised ORs), as encryption or decryption is processed at every OR, it is difficult to link any two links (a link is a connection between two Onion routers) to the same route. Therefore, for a communication going through the Onion routers, the entry OR and exit OR are unlinkable. When there are a large number of connections, it is difficult to find out the two communication ends for any connection that applies Onion routing.

To avoid that the change of "Onion" size in the route built-up stage may give adversary hint about routing information, an "Onion" has to be padded when part of its information has been read and removed, so that the length of the "Onion" keeps the same and it is difficult for an inside observer to obtain the routing information. Refer to [10], if the maximum number of Onion routers in a private route is N, the OP will construct a message of N "Onions" to build an Onion route. When an router receives the "Onion"s, it decrypts all the "Onion"s and obtain the routing information only from the first one. It then adds a dummy packet at the end, and forward the "Onion"s further.

For example, if the maximum hop count N is 5, and the private route is as $OP \rightarrow R_1 \rightarrow R_2 \rightarrow R_3$, the message flow and the messages sent at each router are as follows:

[2]this adversary model is realistic as it may be implemented by having multiple eavesdropping nodes distributed in a Mesh network and combine their observations together.

$$
\begin{aligned}
OP \rightarrow R_1 \quad &: \quad E_{k_{pR_1}}(R_2, k_1), E_{k_{pR_1}}(E_{k_{pR_2}}(R_3, k_2)), \\
&\quad E_{k_{pR_1}}(E_{k_{pR_2}}(E_{k_{pR_3}}(exit, k_3))), \\
&\quad dummy, dummy \\
R_1 \rightarrow R_2 \quad &: \quad E_{k_{pR_2}}(R_3, k_2), E_{k_{pR_2}}(E_{k_{pR_3}}(exit, k_3)), \\
&\quad dummy, dummy, dummy \\
R_2 \rightarrow R_3 \quad &: \quad E_{k_{pR_3}}(exit, k_3), \\
&\quad dummy, dummy, dummy, dummy
\end{aligned} \tag{2}
$$

3.2 Challenges of Applying Onion routing for Mesh Privacy

In our investigated Mesh network, as the Gateway router decides the route, and it has the public keys of all the Mesh nodes, it can act as an OP that selects a private route. However, traditional Onion routing cannot be adopted directly to the Mesh network due to the following reasons:

- Onion routing does not hide a communication end. It achieves anonymity by making communication ends unlinkable. In the Mesh network, one of the communication ends is the Gateway router. If the other communication end, i.e., the Mesh node, is known, the communication privacy is cracked. Therefore, to achieve communication anonymity in the Mesh network, the active Mesh nodes have to be hidden. The privacy achieved by Onion routing is not exact what is needed in the Mesh network.

- Onion routing is vulnerable to a global adversary. As all the connections within the Mesh network is wireless, eavesdropper have better opportunity to conduct traffic monitoring and analysis. In addition, the network size (i.e., in terms of Mesh nodes) normally is not very large. It is possible that there exists a global adversary. Onion routing then is not strong enough to protect Mesh network privacy.

4. REDUNDANT ONION ROUTING

Traditional Onion routing may not be efficient to preserve privacy for a Mesh node because 1) it reveals the communication ends; and 2) it cannot defend against global attacker. In this section, we design a protocol that jointly use Onion routing and communication redundancy, and therefore hides communication ends.

4.1 Routing with a Decoy: A Basic Redundant Onion Routing Protocol

We define traffic moving from the Mesh node toward the Gateway router as *uplink* traffic, while the traffic moving from the Gateway router to the Mesh node as *downlink* traffic. We further call a connection between a Mesh node and the Gateway router a *Mesh network connection*, and a connection between the Gateway router and an Internet address an *Internet connection*.

When a *session initiator*, which in this paper is defined as a Mesh node who would like to connect to the Internet, wants to start a private communication, it sends an access request to the Gateway router. The Gateway router selects a route [3], and uses the shared keys between itself and Mesh nodes in the route to construct an "Onion", and delivers the "Onion" toward the initiator. After a route has been built, the initiator can exchange data with the Gateway router and becomes an *active node*.

The symmetric keys, instead of public keys, are used for reducing the computing cost. Note that the symmetric keys can also be used to verify that an "Onion" is generated by the Gateway router. A signature from the Gateway router is not required. Therefore, using symmetric keys can also mitigate the DoS attack caused by malicious nodes injecting a large number of fake "Onions". The symmetric key between a Mesh node and the Gateway router can be established at the network configuration stage, i.e., when the Gateway router authenticates a newly joining Mesh node.

The Onion structure and the operations for the route built-up are similar to those in (2), except for that the symmetric keys are used. The Mesh nodes in the route and the initiator decrypt the "Onion"s to obtain the assigned session keys and the routing information (i.e., the previous hop and next hop). After the route is built, the session keys will be used by the Gateway router to construct downlink "Onion"s by encrypting downlink data packets. A Mesh node decrypts a downlink "Onion" and forwards the new "Onion" to its next hop that is close to the session initiator. On the other hand, upon receiving an uplink "Onion", a Mesh node in the route encrypts it and then forwards the new "Onion" to its next hop that is closer to the Gateway router. The Gateway router uses the session keys to decrypt the uplink "Onion" and obtain the raw data.

The ID of a session initiator is not carried in a private route. To other Mesh nodes, the initiator is kept anonymous. Although the ID of the initiator is revealed to the Mesh node that is next to it, that node does not know whether it is the initiator or just another forwarder. In addition, the "Onion" structure prevents both inside and outside attackers from comparing packet formats for route derivation.

However, if a session lasts long enough, it is not difficult for an outside attacker that is even not a global adversary to trace down from the Gateway router to an active node according to the link layer packet headers. To address this problem, once the Gateway router receives an access request, a private route should have redundant hops. It dose not end at the session initiator, but extends from there for a few extra hops. For downlink traffic, after the session initiator receives the data for it, it generates a dummy packet as a fake "Onion" and forwards it to the next hop. This next hop decrypts the "Onion" and forwards it as well. The redundant forwarding goes on until a fake "Onion" reaches the end of the route. In this paper we refer the end of the redundant rout as to a *decoy*. A private route generated by redundant routing is illustrated in Fig. 2. F_4 is the decoy.

To avoid that an attacker can discover an active node by monitoring the uplink traffic, in the redundant routing, the active node does not send an uplink traffic directly. Instead, it waits for a so-called *uplink carrier* generated by the decoy. The carrier is sent by the decoy toward the Gateway router. Each Mesh node that receives the carrier encrypts it and forwards it. Once the active node receives the carrier, it replaces it with an uplink packet. It then encrypts

[3]The route selection can be complex if the current network traffic is considered. Route optimization based on traffic and resource usage is beyond the scope of this paper.

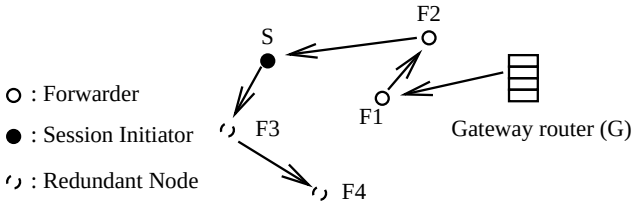


Figure 2: Using redundant route to enhance anonymity.

the packet and forwards the generated "Onion" toward the Gateway Router.

How frequently the decoy sends an uplink carrier depends on the active node's uplink traffic. A session initiator estimates its future uplink traffic and carries the information in the access request. During a session, an active node can modify such information in its previous uplink data packets. The Gateway router decides the frequency for sending an uplink carrier. It then makes the decision as the inner part of an "Onion" that is encrypted by the session keys of all the Mesh nodes in the route. This "Onion" is destined for the decoy.

When the redundant Onion routing is used, for an outside attacker, although it can still trace down to the end of the route, it only knows the session initiator is one of the Mesh nodes in the route. The number of Mesh nodes in the route determines the size of the anonymity set. The anonymity increases when the number gets larger, at the cost of increased communication and computing overhead.

4.2 Privacy Weakness: Not Against Global Adversary

A privacy weakness is that if the transmission of the access request from a session initiator is observed by the adversary, the initiator can be discovered. An access request message may be received by an inside attacker if it is in the route, because Traditional routing protocols [16, 17] or the enhanced secure protocols [18, 19] for multi-hop packet delivery that may be used for access request delivery require that the source ID be revealed in the route. This makes the initiator known to all the nodes in the path. Even if anonymous routing is used[4], an initiator can still be detected in some special cases. Suppose a malicious node that knows the network topology receives an access request from a previous hop, and knows the previous hop is located at the edge of the network. Then to the malicious node, this previous hop is probably a session initiator because it is less likely that other nodes will use it as a forwarder.

Using "Crowds" [11] for access request delivery may mitigate the above problem. However, a global outside adversary, which is also the adversary model in this paper, can catch any message, and therefore discover a session initiator. An example for how an adversary can link a private communication to a Mesh node is as follows. First, a packet transmission from a node is observed by the adversary. After a short time, a private route is observed and that node is included in the route. Even if the adversary does not know whether the packet it intercepted from that node was an access request or not (due to the link layer encryption), based on global transmission information and time correlation, it can make the decision that the Mesh node is probably the session initiator in the private route.

[4]Source can be kept anonymous if routing is based on the link layer trust, i.e., a Mesh node thinks a packet to be legitimate if it is received from its authenticated neighbor.

5. PRIVATE ONION RING: AN APPROACH AGAINST GLOBAL ADVERSARY

To defend against a global adversary, we further explore routing redundancy for node anonymity, by having the decoy be the Gateway router itself. Therefore, any communication starts from the Gateway router and ends at the Gateway router. As the route is a ring, and Onion structure is used for data packet, we refer such a private route as to an Onion ring.

An Onion ring is shown in Fig. 3. The Gateway router builds a ring that consists of n nodes. To any party, e.g., a party in the ring or a party out of the ring, the traffic appears to be sent in the same direction, which is as $G \rightarrow F_1 \rightarrow F_2 ... F_i ... \rightarrow F_n \rightarrow G$. F_i is the ith Mesh node in the ring. G is the Gateway router.

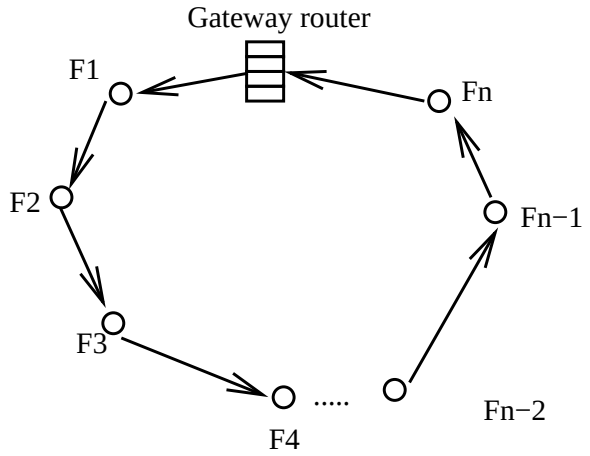


Figure 3: Private "Onion" ring.

In this section, we first present how rings are generated and maintained. We then explain how any member in a generated ring can communicate with the Gateway router. Finally we analyze the anonymity, followed by the discussion of network performance.

5.1 Ring Management

To build an Onion ring, the Gateway router first sends a *ring initiation* message, i.e., a message consists of a number of onions. The message formats and the process at each node in the ring are similar to those in (2). We only show the message at the Gateway router as follows:

$$E_{k_{GF_1}}(RI, F_2, k_1), E_{k_{GF_1}}(E_{k_{GF_2}}(RI, F_3, k_2)), ..., \quad (3)$$
$$E_{k_{GF_1}}(E_{k_{GF_2}}(E_{k_{GF_3}}(...E_{k_{GF_N}}(RI, G, k_n)...)),$$
$$dummy, dummy, ...$$

A Mesh node in the ring, e.g., F_1, uses the pre-assigned symmetric key shared between itself and the Gateway router, e.g. k_{GF_1}, to peel all the "Onion"s and obtain ring information from the first one. The information includes which

ring it is in, which node is its next hop, and what session key should be used. RI stands for the ring ID, which is necessary for a Mesh node to identify rings if it is included in a number of rings. k_1 stands for the session key used by F_1 for data"Onion" operations. The number of nodes included in a ring is determined by the minimum anonymity requirement.

For operating simplicity, rings can be fixed, i.e., once a ring is built, it will be used for a long time. However, this excludes the possibility that the Gateway router can balance the anonymity and network QoS requirements, by changing the ring size. In addition, in case a ring is broken because one of its nodes is naturally down due to, e.g., mis-functions of Mesh equipments, a new ring has to be generated. The intruder identification scheme in 6 also requires change of rings. Based on the above, rings should be dynamic, i.e., new rings may be generated to take place of old rings. A new ring is constructed by Gateway router, which later informs the Mesh nodes included in that ring by sending a ring initiation message. The operations at each Mesh node include decryption and encryption, receiving and transmitting data, and forwarding.

5.2 Ring Communications

We define an *active* ring as the rings where there is at least one active node. We also define a *Mesh transmission cycle* as the entire process that a message is sent from the Gateway Router, operated by the Mesh nodes in the ring, and finally received by the Gateway Router.

For each ring, active or inactive, the Gateway router sends a packet called *request carrier* (i.e., a dummy packet) periodically. The carrier is received by Mesh nodes, in the sequence as how ring is made up. If a node does not have any data to send, it simply encrypts the carrier using the symmetric key shared between itself and the Gateway router, and sends the encrypted message to its next hop. For a node that wants to initiate a session and send data to the Gateway router, it replaces the received carrier with an access request, encrypts the request with its session key, and sends it to the next hop. The request will be received and encrypted by the nodes in the rest of the ring, and finally received by the Gateway router.

For example, if node i in the route sends an access request using the requester carrier, the message flow and the corresponding messages sent at each node in the ring are as follows:

$$\begin{aligned}
G \rightarrow F_1 &: \{RI, seq, req\}, dummy \\
F_1 \rightarrow F_2 &: \{RI, seq, req\}, E_{k_1}(dummy) \\
F_2 \rightarrow F_3 &: \{RI, seq, req\}, E_{k_2}(E_{k_1}(dummy)) \\
... &: ... \\
F_{i-1} \rightarrow F_i &: \{RI, seq, req\}, \\
&\quad E_{k_{i-1}}(...E_{k_2}(E_{k_1}(dummy))...) \\
F_i \rightarrow F_{i+1} &: \{RI, seq, req\}, E_{k_i}(request) \\
F_{i+1} \rightarrow F_{i+2} &: \{RI, seq, req\}, E_{k_{i+1}}(E_{k_i}(request)) \\
... &: ... \\
F_{n-1} \rightarrow F_n &: \{RI, seq, req\}, \\
&\quad E_{k_{n-1}}(...E_{k_{i+1}}(E_{k_i}(request))...) \\
F_n \rightarrow G &: \{RI, seq, req\}, \\
&\quad E_{k_n}(E_{k_{n-1}}(...E_{k_{i+1}}(E_{k_i}(request))...)).
\end{aligned}$$

$\{RI, seq, req\}$ is the packet header. req indicates that message is for sending an access request. seq is the universal packet sequence number. It increases by one whenever there is a packet sent to the ring.

When F_i receives the carrier, it generates a request message req, encrypts it using k_i, and sends the encrypted message with the header to its next hop. A req is further formatted as $\{F_i, addr, traff, QoS\}$. F_i is the identity of the requester. $addr$ is the Internet address it would like to visit. $traff$ and QoS are the estimated traffic pattern and QoS requirement, based on which the Gateway router determines the bandwidth assigned to F_i.

After the Gateway router receives the returning carrier packet, it decrypts the packet using the session keys assigned to the Mesh nodes in the ring, so as to read the access request.

It may happen that two initiators use the same request carrier to send their requests. In that case the one closer to the end of the ring will always erase the previous request and gets granted. The one whose request is erased cannot get a request reply at that transmission cycle, and can only re-send the request when the next carrier comes.

If an access request is granted, the Gateway router will construct an "Onion" for the downlink traffic, and send the "Onion" (i.e., the encrypted downlink data) along with the header to the ring. Each node in the ring decrypts the "Onion", and forwards the new "Onion" and the header to the next hop. The message flow from the Gateway router to F_i and the corresponding formats for the "Onions" at different nodes from G to F_{i-1} are as follows:

$$\begin{aligned}
G \rightarrow F_1 &: \{RI, seq\}, \\
&\quad E_{k_1}(E_{k_2}(...E_{k_i}(F_i, downlinkdata)...)) \\
F_1 \rightarrow F_2 &: \{RI, seq\}, E_{k_2}(...E_{k_i}(F_i, downlinkdata)...) \\
... &: ... \\
F_{i-1} \rightarrow F_i &: \{RI, seq\}, E_{k_i}(F_i, downlinkdata).
\end{aligned}$$

F_i decrypts the "Onion" part, learns that the data is for itself from the ID information, and gets the downlink data. If the data is received correctly, it generates an acknowledgment. If it has uplink data to send, it piggybacks the uplink data to the acknowledgment. Otherwise it pads the packet with dummy data. It then decrypts both the acknowledgment and the data part to generate an "Onion", and sends this "Onion" to its next hop. The "Onion" is decrypted then forwarded by the Mesh nodes in the rest of the ring sequentially, and finally received by the Gateway router. The message flow from F_i to the Gateway router and the corresponding formats for uplink data at different nodes are as follows:

$$\begin{aligned}
F_i \rightarrow F_{i+1} &: \{RI, seq\}, E_{k_i}^{-1}(F_i, ack, uplinkdata) \\
F_{i+1} \rightarrow F_{i+2} &: \{RI, seq\}, E_{k_{i+1}}^{-1}(E_{k_i}^{-1}(F_i, ack, uplinkdata)) \\
... &: ... \\
F_n \rightarrow G &: \{RI, seq\}, \\
&\quad E_{k_n}^{-1}(...E_{k_{i+1}}^{-1}(E_{k_i}^{-1}(F_i, ack, uplinkdata))...).
\end{aligned}$$

The Gateway router then uses the session keys to encrypts the "Onion", so as to get the acknowledgment and the uplink traffic, if any. It is possible that for a while there is no downlink traffic for F_i yet it has uplink traffic to send. In this case, the Gateway router sends a dummy packet working as a *uplink carrier*, in the format of:

$\{RI, seq\}, E_{k_1}(E_{k_2}(...E_{k_i}(ID, uplinkcarrier, dummy)...))$.

The uplink carrier is operated in the same way as for a downlink packet. Once F_i receives the carrier, it can send an uplink data packet. Whether and when the Gateway router should send an uplink carrier to F_i depends on the time that the Gateway router receives the last uplink packet from F_i, and the required time interval between any two consecutive uplink packets for F_i. This time interval is determined by the network traffic and F_i's QoS requirement.

5.3 Anonymity

In the private Onion ring, to an outside observer, an active node behaves exactly the same as other nodes in the ring. The block cipher makes it impossible for a global adversary to discover an active node by comparing the entry and exit traffic at these Mesh nodes. The activity for an active node, such as sending a request, receiving a downlink packet, and sending an uplink packet, is indistinguishable from the activity for an inactive node, i.e., forwarding an "Onion". The link layer encryption between two neighboring Mesh components further improve the anonymity. The anonymity set for any active node in an Onion ring is all the Mesh nodes in the ring. As a large ring achieves a high anonymity, yet results in a low throughput, the ring size can be kept large to improve anonymity only when the network has a low load. On the other hand, if the network is heavily loaded, ring size should be kept small if a communication QoS is required. Only low privacy can be preserved.

When the traffic in the Mesh network is extremely low, e.g., there are only a few active nodes in the network, a ring can include all the Mesh nodes in the network, which in this paper we refer to as a *Big Ring* approach. Big Ring achieves the highest anonymity because the anonymity set includes every Mesh node. Compared to the anonymity approach that broadcasts a packet from Gateway router to the entire Mesh network, the Big Ring approach preserves the anonymity for not only the nodes receiving downlink traffic, but the nodes sending uplink traffic as well.

As rings are dynamic, i.e., rings may change from time to time, a major concern is that a careless ring re-construction may incur an intersection attack based on user profiles. An example of intersection attack is as follows. Assume that a Mesh node initiates a session to connect to an Internet address through a ring. Later it is included in new ring, through which it visits the same address again. Both visits are observed by the adversary that monitors the Gateway router. If the address only has very special visitors, based on the observations, the adversary may conclude that the session initiator is one of the Mesh nodes that are in both rings. The corresponding anonymity set becomes small. Another important profile that is useful for the adversary to conduct intersection attack is the time that a Mesh node connects to the Internet. For example, if a Mesh node usually connects to the Internet at $2am$ in the morning, then this Mesh node probably is one of the nodes that are in the intersected part of the rings generated at $2am$ in different days. To mitigate intersection attacks, traditional K-anonymity anonymous approach [21] or newly proposed "L-diversity" [22] can be considered, under which rings are constructed according to user profiles.

A global aggressive attacker can also jam Mesh nodes one by one to discover the active nodes. Note that it is not difficult for the Gateway router to detect such an attack because of the abnormal node behavior pattern. To maintain privacy, even if the attacker has successfully jammed an active node, the Gateway router will still generate a new ring that consists of a number of nodes, and injects some dummy traffic into it. The injected traffic can have a similar traffic pattern as that in the broken ring. Therefore, the attacker cannot determine whether it has jammed the active node. Node anonymity can still be preserved.

5.4 Network Performance

Onion ring achieves Mesh node anonymity at the cost of communication overhead. Compared with traditional end-to-end paths of which Mesh nodes use random access technique, the performance degradation in ring may not be as severe as at the first look.

An important performance metric is throughput. If the number of hops in an Onion ring is the same as that of an end-to-end route (e.g., a redundant Onion route), the Onion ring has a better throughput. This is partially because in Onion ring, a transmission cycle is for both uplink and downlink traffic, while in redundant Onion routing, uplink traffic has to be transmitted separately. In addition, as all the traffic in Onion rings starts from a Gateway router, the Gateway router can arrange a global scheduling to optimize the network performance, by fully use the wireless bandwidth.

In most cases, the number of hops in a ring is greater than that for an end-to-end connection that uses the shortest path. We may roughly compare the maximum throughput for a node when it uses ring to connect to the Gateway router, to the maximum throughput when it uses the shortest path connection. In Fig. 4 we show some typical cases how rings are built, and how an active node connects to the Gateway router through these rings (circles and ovals) and through direct end-to-end connections (lines) respectively. The ratio between the number of hops needed in a ring and in an shortest path is approximately the same as the ratio between the geographic length of the two routes. It is not difficult to find out the ratios for number of hops for case 1, 2, and 3 are π^2, 8, and 4 respectively.

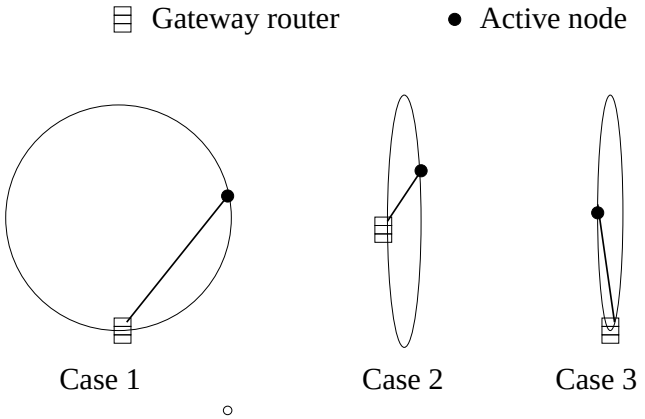


Figure 4: Geographic length for a ring route and end-to-end route.

We assume only one channel is used. For end-to-end connections, we assume CSMA/CA is used for channel access. We consider a simple network scenario where any two Mesh

nodes are within the co-channel interference range with each other. The maximum throughput in the routes then depends on how many hops are required. If when using ring topology, the bandwidth can be used 100% efficiently thanks to the global control, while when using CSMA in the end-to-end communication, it is well known that the channel efficiency is around 60%[5], the ratio for the maximum achievable throughput between a ring and an end-to-end routes in the cases in Fig. 4 then are 5.9, 4.8, and 2.4 respectively. In particular, if the ring can be constructed as in case 3, the throughput degradation is not significant.

Another performance metric is how long it takes a Mesh node to access the Gateway router. A session initiator may suffer a delay because it has to wait for a request carrier. The delay can be reduced by sending out a request carrier more frequently. The bandwidth consumption for sending request carriers can be ignored if the sending frequency is not too high. Another factor that causes access delay is the request collision. A failed Mesh node has to wait for the next request carrier to re-send the access request. However, it is rare that two Mesh nodes start sessions at the same Mesh transmission cycle. Therefore, the probability of a request collision is small. The access delay caused by the request collision is not a significant problem.

Finally, a global scheduling at the Gateway router can guarantee a small delay jitter, which is the most important QoS requirement for real-time services.

A more detailed study on Onion ring performance is our on-going work and will be present in another paper.

6. IDENTIFYING ACTIVE INSIDE ATTACKERS

One of the concerns in Onion ring is that an active inside attacker, i.e., a malicious node in the ring, may drop, manipulate, or replay the "Onion"s, or inject traffic in the ring, to interrupt the ring functions. The following ring re-construction required by reliable communication in the Mesh network may give a global attacker more information to identify active nodes.

The integrity check proposed in [10] can be used for preventing downlink traffic manipulation. Knowing session keys for all the nodes in the ring, the Gateway router is able to construct a message authentication code (MAC) so that a Mesh node in the ring is able to detect whether its previous hop has modified the "Onion". However, such a technique can not be used for uplink traffic because the uplink traffic is generated by a Mesh node, who does not know the session keys for other nodes. In addition, there is no cryptographic way to prevent packet dropping or injection.

We propose an intruder identification scheme, under which the Gateway router can discover the malicious nodes. We focus on the identification of a packet manipulator. The identification methods for other types of inside attackers are similar. The Onion structure and the special communication pattern in an Onion ring (i.e., every packet starts and ends at the Gateway router) make both the intrusion detection and intruder identification less complex.

As we have assumed a perfect link layer error control, a discovery of any erroneous packet indicates an intrusion detection. For a downlink transmission, the Gateway router is aware of an intrusion if it does not receive the acknowledgment from the destined node, which means either the node has not received the packet correctly or the acknowledgment message has been modified. For uplink traffic, the Gateway router detects a packet manipulation if it receives a packet that 1) is not from any node in the ring; and 2) the packet does not match the "Onion" generated by encrypting the uplink carrier by the session keys of all the nodes in the ring.

Once a manipulation is detected, the malicious node has to be identified. To do so, each Mesh node in the ring has to keep the hash results for a number of packets they have previously received. The Gateway router can identify bad links by having the Mesh nodes to show the hash values. For example, referring to Fig. 3, assume F_2 is the malicious node and has modified the "Onion". Required by the Gateway router, it can either send the hash result for the wrong packet which was sent to the next hop F_3, or send the hash value of the right packet that was received from the previous hop F_1. For the first case, the Gateway router will make the decision that link F_1F_2 is questionable, because other than F_2, F_1 also may have modified the "Onion". For the second case, link F_2F_3 is questionable, because it may be F_3 who modified the "Onion". The method in [20] can be applied directly for facilitating a bad link detection in the ring.

If a signed acknowledgment is used for each packet transmission at link layer, in the above example, F_2 can be identified immediately. In the case that the Gateway router judge link F_1F_2 to be questionable, F_1 can send the Gateway router the acknowledgment signed by F_2, which indicates that F_2 has received the correct packet from F_1. When link F_2F_3 is questionable, F_2 is not able to prove that it has sent the right packet to F_3, because the acknowledgment from F_3 is based on the packet that F_2 has modified. In both cases, F_2 can be judged to be malicious.

However, signing acknowledgment for every link layer transmission is expensive, because a large number of public key processes are involved. This leads to the increased computing overhead and packet processing delay at each node. Note that a longer processing delay means a lower throughput in the ring as well. If signature is not used for acknowledgments due to the above concern, the Gateway router can generate new rings and separate the two ends of a questionable link into different rings. A malicious node can be identified if it appears in questionable links many times. The probability of a false positive for the intruder identification scheme depends on how aggressively the Gateway router judges a node to be malicious, i.e., based on how many times a node is in a questionable link can this node be judged malicious.

To identify the malicious Mesh node correctly, reliable communication is required for the Gateway router to obtain hash values from all the nodes in the ring. As the Gateway router knows the network topology, it can build an "Onion"ed request and send it to the tested node. The Onion structure authenticates the Gateway router and the tested node with each other, and keeps the information confidential as well. The route should not include any node that is in the questionable ring.

It is possible that other malicious nodes can be included in this route, who may drop or modify the testing "Onion" or the response. In this case, the Gateway router may not

[5]This result is for single-hop connections. The channel efficiency when using CSMA in multi-hop connections can be worse.

get the required hash value. However, because the route for fetching hash values can be very short and the number of nodes in the route is small, if a malicious node in such a route conducts any attack, it is hidden within a very small group. It may not be difficult for the Gateway router to identify the malicious node with other information or node behavior records. On the other hand, if there is an attack and the Gateway node cannot receive the required hash information, it can still communicate with the tested Mesh node through some other routes.

7. RELATED WORK

One anonymity approach that has been proposed for wired networks is to hide in a group of entities that have similar characteristics. Examples schemes include Mix [7] and Crowds [11]. When the idea of "mixing" is applied to emails, a trusted party "mixes" a group of e-mails with some dummy messages before any email is delivered to its recipients. After the mixing, the inputs and outputs are not linkable. Tarzan [23] adopt the mix approach in peer-to-peer networks. Here a mix relay is a volunteer peer, and peers communicate through a sequence of mix relays. Tarzan therefore does not require any centralized component. Crowds is designed to achieve anonymity for web service requesters. However, Crowds does not provide sender anonymity against a local eavesdropper. Other approaches such as APFS (Anonymous Peer-to-Peer File Sharing) [12] and P^5 [13] use multi-cast or broadcast to achieve both the sender anonymity and receiver anonymity.

In Onion routing [8], a group of users connect to the Internet through the onion routers. For each user, an onion proxy determines an onion route that consists of a number of onion routers. Each onion router encrypts or decrypts the packet it forwards, according to the traffic directions. As a result, it is difficult to link an input of the Onion with an output, because at each onion router the input and output is different to outside observers. In addition, an onion router only knows the previous hop and next hop of a message in an onion. The second generation of onion routing tool is called Tor. According to [9], Tor addresses limitations in the original design by adding perfect forward secrecy, congestion control, directory servers, integrity checking, configurable exit policies, and a practical design for location-hidden services via rendezvous points. However, tor is not designed to defend against global attacks. A passive global observer may be able to link an input flow with an output flow by traffic analysis. In [24], it has been showed that an active attacker can conduct traffic analysis attack by using onion routers as an oracle to infer the traffic load on the remote node, even though the attacker sees only partial information of the network. Another work [10] proposes a padding scheme using random series, as well as an integrity algorithm that prevents any onion router from manipulating "Onioned" packet.

Privacy has not been extensively investigated in Mesh networks, yet it has been studied in another multi-hop wireless network, the ad hoc network, under the assumption that the existence of a global attacker is impossible. Anonymous communication in ad hoc network has been studied in [25], which proposes a novel untraceable on-demand routing protocol. Onion structure is used only for route discovery. To reduce the cost and latency of the encryption/decryption, symmetric key based *Boomerang Onions* are used. Once a route is found, pseudo-random numbers are used as temporary IDs for the nodes along the route. In [26], a neighborhood authentication protocol that allows neighboring nodes to authenticate each other without revealing their identities is designed for communication anonymity. However, in this approach, the destination ID has to be revealed for on-demand route discovery. Therefore, only conditional anonymity can be achieved for the destination. A tracer knows which node is the destination, yet the tracer does not know where the destination is. In [27], a geo-casting approach is used to achieve destination anonymity. A packet for a destination is locally flooded within a region, known as an anonymity zone, where the destination is located. The anonymity set includes all the nodes located in the anonymity zone. The size of an anonymity set may decrease due to node mobility, however the approach simplifies the group management in mobile ad hoc networks. In another work [28], node positions are used as pseudonyms for communication anonymity.

A similar anonymous approach that uses ring topology is DC-Net [29], an algorithm designed by Chaum. DC-Net uses shared secret keys among the parties within the same anonymity set, to hide the source of a transmitting message. The differences between DC-Net and our design are as follows. First, other than using transmission redundancy, DC-Net uses message redundancy as well, e.g., a message is padded with a lot of dummy bits, of which the length is normally a few times longer than the length of the message itself. In Onion ring, there is no message redundancy. Second, DC-Net requires shared secret keys among peer participants. A leakage of key information (e.g., some participants may be malicious or compromised) may lead to privacy degradation. In addition, there is scalability problem for rekeying when compromised peers are discovered and have to be removed. In Onion ring, participants share keys with a trusted center. The possibility of privacy degradation caused by key leakage is smaller. It is also easy to generate a new ring that excludes any compromised parties. Finally, Detecting any malicious participants in DC-Net, e.g., a jammer, requires more complex cryptography and extra computation that is at least in the order of n^2 (refer to [30]). In Onion ring, basic cryptography is used, and only a few polling messages are needed for malicious participant detection.

8. CONCLUSION AND FUTURE WORKS

This paper presents the Onion Ring protocol that achieves anonymity in a Mesh network. The Onion structure protects the routing information from inside attackers. The ring topology and passive session initiation make the anonymity set the same as the number of nodes in the ring. Even to a global adversary, the behavior of each node in an onion ring is un-distinguishable. We also discuss the computing and communication overhead of this approach.

The future work will be simulation study on the routing performance of the Onion ring. In particular, the performance will be compared with those using the shortest path and redundant routing. Another future work will be the detailed design on routing optimization and packet transmission scheduling, given that the Gateway router works as a central control that has the global network information. The future work will also include intelligent generation of the rings, i.e., generating rings according to the profiles of the Mesh nodes to mitigate intersection attacks.

9. REFERENCES

[1] R. Draves, J. Padhye, and B. Zill, *Routing in multi-radio, multi-hop wireless mesh networks*, In Proceedings of the 10th annual international conference on Mobile computing and networking, 2004.

[2] J. Jangeun and M.L. Sichitiu, *The nominal capacity of wireless mesh networks*, In IEEE Wireless Communications, volume 10 of 5, pages 8-14, Oct. 2003.

[3] A. Raniwala, K. Gopalan, and T. Chiueh, *Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks*, SIGMOBILE Mobile Comput. Commun. Rev., 8(2), 2004.

[4] K. Rayner, *Mesh wireless networking*, Communications Engineer, 1(5):44 47, Oct.-Nov. 2003.

[5] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris *Link-level Measurements from an 802.11b Mesh Network*, in SIGCOMM 2004.

[6] *IEEE Std 802.11b-1999*, url = http://standards.ieee.org/getieee802/802.11.html.

[7] D. L. Chaum, *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms*, Communications of the ACM, 24(2):84–88, 1981.

[8] M. Reed, P. Syverson, and D. Goldschlag, *Anonymous Connections and Onion Routing*, IEEE Journal on Selected Areas in Communication Special Issue on Copyright and Privacy Protection, 1998.

[9] R. Dingledine, N. Mathewson and P. Syverson, *Tor: The Second-Generation Onion Router*, in 13th USENIX Security Symposium, 2004.

[10] J. Camenisch and A. Lysyanskaya, *A Formal Treatment of Onion Routing*, in Proceedings of CRYPTO, 2005.

[11] M. K. Reiter and A. D. Rubin, *Crowds: Anonymity ForWeb Transactions*, ACM Transactions on Information and System Security, 1(1):6–92, 1998.

[12] V. Scarlata, B. Levine, and C. Shields, *Responder Anonymity and Anonymous Peer-to-Peer File Sharing*, IEEE International Conference on Network Protocols (ICNP), Riverside, CA, 2001.

[13] R. Sherwood, B. Bhattacharjee, and A. Srinivasan, *p5: A Protocol for Scalable Anonymous Communication*, IEEE Symposium on Security and Privacy, pages 53–65, Oakland, CA, May 2002.

[14] *Routing Information Protocol*, RFC, url=http://www.faqs.org/rfcs/rfc1058.html.

[15] *OPSF Version 2*, RFC, url = http://www.faqs.org/rfcs/rfc2328.html.

[16] C.E. Perkins and E.M. Royer, *Ad-hoc On-Demand Distance Vector Routing*, in Proc. of the 2^{nd} IEEE WMCSA, pp.90–100, 1999.

[17] D. Johnson and D. Maltz, *Dynamic Source Routing in Ad Hoc Wireless Networks*, in Proc. of ACM SIGCOMM, 1996.

[18] Y.-C. Hu, D. B. Johnson, and A. Perrig, *SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks*, in Proc. of the 4^{th} IEEE WMCSA 2002, June 2002, pp. 3-13. Submitted for publication.

[19] Y.-C. Hu, D. B. Johnson, and A. Perrig, *Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks*, in Proc. of MOBICOM'02, 2002.

[20] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens, *An On-Demand Secure Routing Protocol Resilient to Byzantine Failures*, In ACM Workshop on Wireless Security (WiSe), 2002.

[21] L. Sweeney, *K-Anonymity: A Model for Protecting Privacy*, International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10(5):557–570, 2002.

[22] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, *l-Diversity: Privacy Beyond k-Anonymity*,

[23] Michael J. Freedman, Robert Morris, *A Peer-to-Peer Anonymizing Network Layer*, in Proceedings of ACM Conference on Computer and Communication Security, 2002.

[24] S. J. Murdoch and G. Danezis, *Low-Cost Traffic Analysis of Tor*, in IEEE Symposium on Security and Privacy, 2005.

[25] J. Kong and X. Hong, *ANODR: Anonymous on Demand Routing with Untraceable Routes for Mobile Ad-hoc Networks*, in Proceedings of ACM international symposium on Mobile ad hoc networking and computing, June 2003.

[26] Y. Zhang, W. Liu, and W. Luo, *Anonymous Communications in Mobile Ad Hoc Networks*, in Proceedings of INFOCOM, 2005.

[27] X. Wu and E. Bertino, *Achieving K-Anonymity in Mobile Ad Hoc Networks*, accepted in workshop on Secure Network Protocols, 2005.

[28] X. Wu and B. Bhargava, *AO2P: Ad hoc On-demand Position-based Private Routing Protocol*, IEEE Transactions on Mobile Computing, Vol. 4, No. 4, Pages 335–348, 2005.

[29] D. Chaum, *The Dining Cryptographers Problem: Unconditional Sender and Receipient Untraceability*, Journal of Cryptography, Vol. 24, No. 2, Pages 65–75, 1988.

[30] P. Golle and A. Juels, *Dining cryptographers revisited* Advances in Cryptology - Eurocrypt 2004.