A clique-based secure admission control scheme for mobile ad hoc networks (MANETs)

Qing Chen a,*, Zubair Md. Fadlullah a, Xiaodong Lin b, Nei Kato a

a Graduate School of Information Sciences, Tohoku University, Sendai 980-8579, Japan
b Faculty of Business and Information Technology, University of Ontario Institute of Technology, Oshawa, Ontario, Canada

Article history:
Received 30 June 2010
Received in revised form 2 November 2010
Accepted 11 December 2010

Keywords:
Mobile ad hoc networks (MANETs)
Key management
Pairing-based cryptography

Abstract

Wireless mobile ad hoc networks (MANETs) do not have centralized infrastructure and it is difficult to provide authentication services. In this paper, we apply Certificate Graph (CG) and identity-based security in designing an admission control scheme for MANETs. We first use one-hop message exchange to build CG at each mobile node. Then we select maximum clique nodes in CG as distributed Certificate Authorities (CAs). We use identity-based key agreement from pairings to protect each session. Then we prove the security by Canetti–Krawczyk (CK) model-based analysis. We demonstrate the effectiveness and feasibility of our protocol through computer simulations.

© 2010 Elsevier Ltd. All rights reserved.

1. Introduction

MANETs show an alternative way of communication, where wireless nodes cooperate with one another to forward packets from a source to an appropriate destination, in a multi-hop fashion. Each MANET node collects information from its one-hop neighbors. Thus, MANETs nodes do not need to know about the whole network topology. In other words, there is no need for a centralized infrastructure in a MANET. This design consideration for MANETs makes a number of differences to the conventional centralized networks, namely (i) dynamic topology (Chen et al., to appear), (ii) resource constraints (He et al., 2010), (iii) lower cost, (iv) limited physical security. These features have their pros and cons in MANETs. For instance, MANET nodes are able to roam freely at certain speed. While this feature allows for sufficient flexibility in deploying a network, it may also lead to unpredictable situations whereby the nodes may move too far away and become isolated. On the other hand, MANET nodes may be notebook computers, cell phones or even PDAs that are usually suffer from relatively short battery life. In addition, MANETs are meant to be deployed swiftly without infrastructures (e.g., base stations or access points), which cuts considerable cost. Finally, MANETs suffer from limited physical security. This is due to the fact that mobile terminals (e.g., notebooks, PDAs, and so forth) usually do not have sufficiently strong secure hardwares as they increase cost and power-consumption. Indeed, the usability and reliability of MANETs strongly depend on its security. However, because of its openness and lack of centralized services, it is still quite an endeavor to create a secure environment for MANETs. The fact remains that it is quite difficult to ensure confidentiality and authenticity in MANETs.

MANET is subjected to a number of attacks, ranging from rather simple to sophisticated ones. For instance, a selfish node in the MANET may not be willing to route packets to others. It may also discard data packets that it received from other nodes. On the other hand, more sophisticated routing attacks against MANETs may disrupt route discovery. Furthermore, they may disrupt the route maintenance by disobeying the routing protocols. Blackhole attack (Aad et al., 2008), Byzantine attack (Awerbuch et al., 2002), wormhole attack (Maheshwari et al., 2007), and spoofing attack (Sanzgiri et al., 2002) are illustrations of various significant threats to MANETs.

For the purpose of secure group communications, cryptography has been incorporated into MANETs. Among the most popular techniques, symmetric (Diffie and Hellman, 1976; Steiner et al., 1996; Bhaskar et al., 2007) and public key infrastructures (Zhou and Haas, 1999; Çapkun et al., 2003; Yi and Kravets, 2003; Rachidi and Benslimane, 2006; Rachidi et al., 2007) are worth mentioning. In the former, two or more users are able to establish secure communications by sharing a common secret key. However, in a MANET with a large number of nodes, private key agreement protocols tend to require numerous message exchanges, which consequently causes significant overheads. Public key infrastructure (PKI) employs a pair of keys to encrypt and/or decrypt the messages. While the private keys need to be kept secret by both parties, the public keys can be widely distributed. In addition, PKI reduces the communication overhead. However, deploying PKI in
MANETs demands for a trusted third party as the Certificate Authority (CA). Providing an online CA presents an enormous technical challenge in a MANET due to its dynamic topology. As a consequence, the public key-based authentication service for the MANET should be both decentralized and autonomous.

To this end, we present a novel PKI-based key management protocol in this paper. The key management protocol ensures secure admission control in MANET environments. In our protocol, we assign the responsibilities of the authenticator to multiple CAs, which are selected from a pool of users with the highest trust levels. In our approach, we do not resort to manual selection of CAs as that in the work of Yi and Kravets (2003). We rather employ a certificate graph (CG) to represent the friendship amongst the participants. We assume that the trust in such CGs can be, indeed, established by utilizing a secure channel as demonstrated previously (Čapkun et al., 2003). Our approach is similar to human social networks in which good (i.e., non-malicious) users are expected to have more friends than bad (i.e., malicious) ones. The most trustworthy subset of these good users in a MANET is represented by the maximum clique and is selected as the authenticator of this group.

We present the remainder of this paper as follows. Section 2 describes several research works based on PKI. Section 3 presents the proposed maximum clique-based key management and secure admission control scheme. In Section 4, we prove the security of the session key exchange. In Section 5, we test the protocol through computer simulations. Finally, Section 6 concludes the paper.

2. Related work

2.1. Security attacks against MANETs

Since a MANET does not require centralized infrastructure, it is more vulnerable to threats in contrast with traditional networks. While there are many threats that may disrupt communications in MANETs, they can be categorized into two groups, namely passive and active attacks. The former includes eavesdropping. The latter includes a variety of malicious threats, from simple selfish behavior to complicated ones such as Byzantine, wormhole, blackhole, and spoofing attacks.

A MANET is designed to be an open network since wireless nodes have to relay data packets in a multi-hop fashion. As a consequence, a node on the routing path can easily capture all the packets. It can then extract confidential information by analyzing those packets. Although this is a passive attack, the eavesdropper can manipulate the captured information to launch active attacks later on (e.g., replaying packet at a later time and impersonating the source node).

Also, each participant in a MANET is supposed to cooperate on route discovery and packet routing in order to establish the communication. However, because mobile terminals have limited electric power, a selfish node may attempt to save its own battery and decide not to forward the packets bound for other nodes. Meanwhile, it may continue to consume the power resources of other nodes in order to send its own packets. While such selfish behavior may not disrupt the communication of the entire network, it may significantly degrade the quality of service.

More sophisticated attacks against MANETs include Byzantine failures and wormhole attacks. Byzantine failures are actually an agreement problem whereby some authenticated yet malicious nodes (i.e., traitor nodes) attempt to send misleading information to trick or confuse the source node. The wormhole attack consists of a diﬀerent scenario comprising two colluding attackers, which manipulate the route discovery of the on-demand shortest path routing protocols. In the route discovery, each attacker broadcasts Route REQuest (RREQ) packets to all its neighbors. One attacker stays close to the source while its accomplice stays close to the destination. These two attackers construct a long distance one-hop link by simply employing directional antenna. As a consequence, the source is unable to take any other route except the wormhole constructed by these attackers, which can then capture or discard all the data packets.

Each secure protocol (Hu et al. 2003a,b,c, 2005) may protect against one or few security threats. However, there lacks a general solution. One way to establish secure communication in MANETs is to provide efficient authentication and certification services. In the remainder of this section, we present the relevant research works.

2.2. Authentication and certification in MANETs

The most popular cryptographic techniques can be categorized into two groups, namely (i) symmetric key infrastructure and (ii) public key infrastructure (PKI). The former establishes common secret keys between the users. Examples of the symmetric key-based cryptography include Diffie–Hellman two-party agreement (Diffie and Hellman, 1976), group Diffie–Hellman (Steiner et al., 1996), asymmetric group Diffie–Hellman (Bhaskar et al., 2007), and so on. The basic idea of Diffie–Hellman two-party agreement protocol is to generate a strong common key based on weak shared secrets. Let us consider two users Alice and Bob. In the Diffie–Hellman two-party agreement approach, Alice generates two random numbers denoted by a and p, which are common secrets. Alice then computes \( a^p \) and sends the result to Bob. Bob, on the other hand, produces a random number q, calculates \( a^{p+q} \), and sends the result back to Alice as the common secret key. In this manner, these two users are able to perform message encryption/decryption by using the common secret key. Protocols in Steiner et al. (1996) and Bhaskar et al. (2007) extend the idea of Diffie–Hellman two-party key agreement to n-party situations that suite for MANET environments. While these schemes may protect communication for MANET nodes, the key agreement procedure can cause significant overhead when the number of nodes increases. This scalability issue occurs due to the bidirectional message exchange in each hop of the communication.

PKI offers a much more effective solution. It relies on asymmetric cryptography and CA to generate a pair of keys (i.e., public and private keys) for each network participant. However, because of the dynamic topology and lack of centralized service, PKI should also be decentralized and self-organized in MANET. As a consequence, it is a challenge to integrate PKI into MANET (Čapkun et al., 2003; Yi and Kravets, 2003). The reason for choosing PKI are as follows as demonstrated by Čapkun et al. (2003) and Yi and Kravets (2003).

- **Confidentiality**: message encrypted with Alice’s public key cannot be decrypted by anyone except the holder of the corresponding private key—Alice herself, which ensures the confidentiality of the message.

- **Authenticity**: message signed with Alice’s private key can be verified by anyone who has access to Alice’s public key, therefore, proving that the message comes from Alice. This ensures the authenticity of the message sender.

Researchers formulated several other approaches based on PKI. Among them, Threshold Cryptography and Distributed Servers, Self-Organized Certificate Chain, and Mobile Certificate Authorities (MOCA) are worth mentioning. Threshold Cryptography and Distributed Servers (Zhou and Haas, 1999) were the first ever protocol to employ PKI to secure ad hoc communication. In this approach, n nodes were chosen from the network as CAs. In a MANET with configuration of \( (n, k+1) \), when a user wants to sign
a certificate, it has to contact \((k+1)\) servers \((n \geq 3k+1)\) to collect \((k+1)\) partial signatures in order to compute a full signature. However, contacting with \((k+1)\) servers means that the MANET will experience significant overhead and communication delay as the value of \(k\) increases. Another shortcoming is the lack of detail on how to choose the appropriate servers.

Self-organized certificate chain (Čapkun et al., 2003) is a fully distributed self-organizing public key management system. Similar to Pretty Good Privacy (PGP), users generate and distribute their own public key certificates. Furthermore, the users themselves are capable of issuing certificates for others. Instead of employing online CAs, each node maintains locally a selected number of certificates according to pre-defined rules. Let us consider the scenario when a user \(v_1\) attempts to verify the authenticity of \(v_2\)'s public key certificate. For this purpose, \(v_1\) asks for and merges \(v_2\)'s stored certificates at first. Then \(v_1\) tries to find a certificate chain from \(v_1\) to \(v_2\) in the merged certificates. In order to do so, the self-organized certificate chain protocol executes as follows.

- **Step 1:** Each user generates his own public/private key pair. If \(v_1\) believes that a public key \(K_v\) does belong to \(v_2\), then it may be able to issue a public key certificate in which \(K_v\) is bound to \(v_2\) by the signature of \(v_1\). There are several reasons for \(v_1\) to believe that \(K_v\) belongs to \(v_2\). For instance, \(v_1\) and \(v_2\) may have exchanged their keys through a dedicated secure channel (e.g., through secure socket layer (SSL)). Every node stores certificates locally, which are (i) certificates that issued by itself; (ii) selected certificates issued by other nodes, according to certain rules. The result of issuing certificates is a certificate graph, denoted by \(G\).
- **Step 2:** Certificate exchange allows users to share their certificates. Each user exchanges its own certificates periodically with one-hop neighbors.
- **Step 3:** In this step, \(v_1\) authenticates the certificate of \(v_2\). When \(v_1\) wants to verify \(v_2\)'s public key, denoted by \(K_v\), \(v_1\) asks for \(v_2\)'s certificate graph \(G_v\). Upon this request, \(v_2\) sends \(K_v\) to \(v_1\). At this point, \(v_1\) merges \(G_v\) with \(G_v\) and attempts to discover a certificate chain from \(K_v\) to \(K_v\) in \(G_v\).

This approach is implemented in such a way that there is no need for any centralized server. Thus, this solution is fully distributed and seems an attractive choice for MANETs. However, it can only guarantee probabilistic certificate requests. The reason is that a user may not store all the certificates required for computing the appropriate chain to verify the authenticity of another user.

Similar to threshold cryptography in Zhou and Haas (1999), the MOCA (Yi and Kravets, 2003) protocol also shares CA responsibilities over a number of manually selected nodes. It is assumed that mobile nodes are different in power, capabilities, transmission range, and physical security. Amongst them, the most powerful and secure nodes are selected as MOCAs. These MOCAs generate public/private keys and store certificates. However, the key authentication procedure in MOCA is similar to the work in Zhou and Haas (1999). In other words, when \(v_1\) wants to authenticate \(v_2\)'s public key, \(v_1\) sends out Certificate REQUEST (CREP) packets. Any MOCA, which received a CREP, responds with a Certificate REPly (CREP) packet containing its partial signature. \(v_1\) waits for a certain period of time for \(k\) such CREPs. Upon receiving \(k\) valid CREPs, \(v_1\) is able to construct a full signature and certificate request is granted. However, the method to choose an appropriate value of \(k\) for networks with different numbers of users presents a substantial problem for MOCA. In fact, the cost of reconfiguration is significantly high in MOCA. Besides, the selection of CAs in MOCA is made upon physical characteristics that may be inadequate for ad hoc networks in particular. In our work, we attempt to address these issues and select CAs by searching for the maximum clique in certificate graphs.

### 3. Proposed protocol

In this section, we introduce a novel key management protocol to perform admission control for MANETs. Inspiration comes from Čapkun et al. (2003) and MOCA (Yi and Kravets, 2003). We combine the benefits of both certificate graphs and CAs. By searching for the maximum clique in certificate graphs we find the most trustworthy nodes. Then we assign CA responsibilities to these clique nodes. The details of our proposed scheme are presented in the remainder of this section.

#### 3.1. Considered trust-based model

For each node \(i (0 < i \leq n)\) in the network, \(i\) keeps a certificate graph (CG) \(G_i\). Each \(G_i\) consists of \(V_i\) and \(E_{jk}\) \((0 < j, k \leq n)\), where \(v_i\) represents node \(i\), while \(E_{jk}\) represents mutual certification between \(v_j\) and \(v_k\) \((v_j, v_k \in E_{jk})\). Thus, for \(n\) nodes, there are \(G = \{G_1, G_2, \ldots, G_n\}\) and \(V = \{v_1, v_2, \ldots, v_n\}\).

Each node runs a maximum clique searching algorithm (Skiena, 1998; Dharwadker, 2006) on CG, denoted by \(G\) of \(V\). The maximum clique searching problem is defined as follows. Given a graph \(G_i = (V_i, E_i)\), the maximum clique is the largest subset \(C \subset G_i\) such that \(\forall_{E_{jk}} \in C, E_{jk} \in G_i\). Given a CG \(G = (V, E)\) (Fig. 1(a)), where

![Fig. 1. Searching for maximum clique in CG. (a) Certificate graph. (b) Maximum clique (in red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)](image)

V = {v₁, v₂... vₙ} and E = {E₁₂, E₂₃... Eₙ₋₁ₙ}, Algorithm 3 outputs clique C₀ of G (Fig. 1(b)).

**Algorithm 1. Subfunction1( ).**

     while ∃Ejk ∈ G, where v_j ∈ C_i ∩ v_k ∈ C_i do
     C_i ← C_i ∪ {v_j, v_k}
     output ← C_i

**Algorithm 2. Subfunction2( ).**

     for ∀v_j ∈ G do
     if ∃Ejk ∈ G, where v_k ≠ C_i ∩ v_j ∈ C_i then
     // Let p denotes the number of edges Ejk
     // where Ejk ∈ G ∧ Ejk ≠ C
     p ← p + 1
     end if
     if p = n – 1 then
     C <- (C – {v_j}) ∪ {v_j}
     Call Subfunction1( ) on C
     else
     output ← C
     end if
     end for

**Algorithm 3. FindMaximumClique( ).**

     for ∀v_j ∈ G do
     C_i ← {v_j}
     Call Subfunction1( ) on C_i
     for r = 1 to k do
     Call Subfunction2( ) on C_i
     end for
     end for
     for ∀C_i, C_j do
     C_j ← C_j ∪ C_i
     Call Subfunction1( ) on C_j
     for r = 1 to k do
     Call Subfunction2( ) on C_j
     end for
     end for

Now, we justify the reasons for choosing the maximum clique as CA in a MANET. First, maximum cliques are found by the MANET nodes themselves. Since they are not manually selected, this ensures a decentralized and autonomous infrastructure suitable for MANET topologies. Second, in a mobile network with more than one CA, it is obvious that every CA should be familiar with the other CAs by knowing their public keys. Besides, certificates stored at different CAs must be consistent. Otherwise, the CAs may present conflicting certificates to one another. In a maximum clique derived from a given MANET topology, each member in the clique knows its other members. In other words, there does not exist two members that are stranger to each other. This ensures close cooperation amongst the CAs. In addition, the considered network may consists of both malicious and non-malicious users. For instance, some malicious users may be selfish and disrupt the packet routing. Similar to social networks, the good (i.e., non-malicious) users in a MANET are trusted by more nodes in the envisioned maximum clique-based approach. On the other hand, the malicious users with lower trust levels are left with zero or few friendly neighbors to communicate with. Thus, by constructing the maximum clique, we actually establish the most trustworthy subset of the good users in the mobile ad hoc network.

### 3.2. Proposed extension to AODV protocol

To integrate admission control scheme into MANET, we extend AODV protocol with our key management function. AODV protocol employs Route Request (RREQ) and Route Reply (RREP) packets for route discovery. When a source node v₁ wants to send a message to v₂, v₁ broadcasts RREQ packets to the network. Neighbors of v₁ that receive this RREQ keep forwarding it to their neighbors, respectively, until the RREQ reaches v₂. v₂ then solicits a RREP and sends it back to v₁. v₁ gets a RREP, creates a routing entry for v₂, and can communicate with v₂ (Fig. 2).

![Route control packet in AODV protocol.](Figure 2)

### 3.3. Protocol implementation

We introduce three additional packets to integrate key management function into AODV protocol. These packets are explained as follows.

- **HELLO packet** contains the public key issued by a user itself;
- **certificate chain packet** contains a chain of certificates that may also be exchanged amongst friendly neighbors (i.e., non-malicious nodes associated with high trust levels);
- **encrypted packet** is used for secure session after CAs are chosen, as discussed in Section 4.

In our proposed scheme, CA selection proceeds as the following three steps:

1. **Issuing certificate.** The protocol begins with issuing of certificates. At this stage, users issue certificates for their trustworthy neighbors, which is similar to Čapkun et al. (2003). The issuing of certificates is bidirectional, which means if a user v₁ issues a certificate for another user v₂, v₂ will also issue a certificate for v₁ such that v₁ and v₂ become friends.

2. **Exchange of certificates.** These certificates can be exchanged amongst the nodes, which are considered as friends. This is achieved by exchanging the certificate chain packets with friends.

3. **Searching for the maximum clique in CG.** The result of issuing and exchanging certificates is a CG. With the CG, users can gain knowledge of their respective neighbors. By searching for the maximum clique in the CG, a user can find a subset of nodes, which are the maximum clique members. These maximum clique members are selected as CAs (as shown in Fig. 1(b)).

### 4. Encrypted session and security analysis

In data communication between any pair of nodes, we encrypt each session with a session secret key. We first describe the...
4.1. Pairing-based key exchange

We use identity-based key exchange protocol (Chen and Kudla, 2003; Pan et al., 2007) from pairing (Boneh and Franklin, 2001) for encryption. Our system parameters are presented in Table 1.

Fig. 3 shows the system model. Each clique CA stores identity/public key pair only for a number of mobile nodes. For example, node $N_a$'s identity/public key is known to $CA_A$ but unknown to $CA_B$. There are two shared resources $Resource_A$, $Resource_B$ in the network. Access to $Resource_A$, $Resource_B$ is controlled by $CA_A$, $CA_B$, respectively. $N_a$ must make request to $CA_A$ in order to gain access to $Resource_B$. Table 2 explains session details.

4.2. Security analysis with CK model

With the CK model, we say that a key exchange protocol is secure if under the allowed adversarial actions it is impossible for the attacker to distinguish the value of a key generated by the protocol from an independent random value.

There are two adversary models defined in the CK model, authenticated-links model (AM) and unauthenticated-links model (UM). In AM, the adversary can pretend to be a legitimate party, but it cannot forge or replay messages from a legitimate party. In UM, the adversary can launch all attacks the AM adversary can do, including forging and replaying messages.

Let $\mathcal{A}$ and $\mathcal{U}$ denote the adversary in AM and UM, respectively. Let $AUTH_{\mathcal{A}}$ denote the global output of all parties running the message-driven protocol $\pi$ in AM. Similarly, let $UNAUTH_{\mathcal{U}}$ denote the global output of all parties running protocol $\pi$ in UM.

**Definition 1.** Let $\pi$ and $\pi'$ be message-driven protocols for $n$ parties. We say that $\pi$ emulates $\pi'$ in unauthenticated networks if for any UM adversary $\mathcal{U}$ there exists an AM adversary $\mathcal{A}$ such that $AUTH_{\mathcal{A}}$ and $UNAUTH_{\mathcal{U}}$ are computationally indistinguishable.

**Definition 2.** A compiler $C$ is an algorithm that takes for input descriptions of protocols and outputs descriptions of protocols. An authenticator is a compiler where for any protocol $\pi$, the protocol $C(\pi)$ emulates $\pi$ in unauthenticated networks.

**Theorem 1.** Let $\lambda$ be an message transmission (MT) authenticator (i.e., $\lambda$ emulates MT in unauthenticated networks), and let $C(\pi)$ be a compiled constructed based on $\lambda$ as described above. Then $C(\pi)$ is an authenticator.

**Definition 3.** We let $\mathcal{U}$ perform a test-session query among the sessions that are completed, unexpired and unexposed at the time. Let $K$ be the corresponding session key. We toss a coin $b$, $b \in \{0, 1\}$. If $b=0$ we provide $\mathcal{U}$ with the value $K$. Otherwise, we provide $\mathcal{U}$ with a value randomly chosen from the probability distribution of keys generated by protocol $\pi$. The adversary $\mathcal{U}$ is allowed for a UM adversary except exposing the test-session.

**Definition 4.** A key exchange protocol $\pi$ is called session key (SK)-secure if the following properties hold for any $\mathcal{U}$ in UM.

1. If two uncorrupted parties complete matching sessions then they both output the same session key;
2. the probability that $\mathcal{U}$ guesses the correct $b$ satisfies $Pr[\mathcal{U}Guess_{b}] \leq \frac{1}{2} + \varepsilon$, where $\varepsilon$ is a negligible fraction.
Assume the probability that $A$ can break the encryption function $E$ or getting secret key $r$. Assume the probability that $A$ can break $E(r)$ is $\epsilon_e$, then the probability that $A$ can get $r$ is $Pr[Guess_k] = \epsilon - \epsilon_1 - \epsilon_2$. Similarly, $CA_2$ sends $E_{K_{N_2,CA_2}}(r)$ to $CA_1$. $A$ can get $r$ by either breaking the encryption function $E(r)$ or getting secret key $r$. Assume the probability that $A$ can break $E(r)$ is $\epsilon_2$, then the probability that $A$ can get $r$ is $Pr[Guess_k] = \epsilon - \epsilon_1 - \epsilon_2$. Given that $CA$ is trustworthy and $N_A$ is not corrupted, $K_{N_2,CA}$ can only be computed from $H(e(P_{N_2},r,P_{CA}))$, which is a BDH problem. Assume $A$ can solve BDH problem with probability $\epsilon_2$ and $\epsilon_2 \geq \epsilon - \epsilon_1 - 2\epsilon_2$. Thus we have $\epsilon \leq \epsilon_1 + \epsilon_2 + \epsilon_3$. If $\epsilon$ is non-negligible, then at least one of $\epsilon_1, \epsilon_2, \epsilon_3$ must be non-negligible, which is a contradiction since $\epsilon_1$ is negligible. So $\epsilon$ must be negligible, and we have $Pr[Guess_k] \leq \frac{1}{2} + \epsilon$. Therefore, protocol is SK-secure in AM.

4.4. Proving key exchange is SK-secure in UM

We use the following authenticators in simulation the message exchange:

- a session key MT authenticator (Boyd et al., 2004) simulates message transmission from $N_A$ to $CA_1$, from $CA_2$ to $CA_1$, from $CA_A$ to $N_A$;
- a encryption MT authenticator (Bellare et al., 1998) simulates message transmission from $CA_A$ to $N_A$.

Since these authenticators are provable secure, by Theorem 2 we know that protocol is SK-secure in UM.

5. Performance evaluation

We run computer simulations in two different scenarios. In the first scenario, we test the protocol in a small scale network to evaluate the success ratio and responsive delays in comparing with MOCA. In the second scenario, we test the protocol in networks of various sizes to evaluate the scalability of the proposed scheme.

Table 2

<table>
<thead>
<tr>
<th>$N_A$</th>
<th>$CA_B$</th>
<th>$CA_A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(ID_{N_A},ID_{CA_A},ID_{N_2},r_p,P_r,P_{E_d}(r))$</td>
<td>$(ID_{CA_A},ID_{CA_2},ID_{N_2},r_p,P_r,P_{E_d}(r))$</td>
<td>$TID \in {0,1} = S_{ID} = S_{A2H_2}(H_2(TID))$</td>
</tr>
<tr>
<td>$(ID_{CA_A},ID_{CA_2},ID_{N_2},r_p,P_{TID},E_{S_{TID}}(r)))$</td>
<td>$(ID_{CA_A},ID_{CA_2},ID_{N_2},r_p,P_{TID},E_{S_{TID}}(r))$</td>
<td>$(ID_{CA_A},ID_{CA_2},ID_{N_2},r_p,P_{TID},E_{S_{TID}}(r))$</td>
</tr>
</tbody>
</table>

Table 3

<table>
<thead>
<tr>
<th>Simulator</th>
<th>QualNet 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routing protocol</td>
<td>AODV</td>
</tr>
<tr>
<td>Scenario dimension</td>
<td>1000 m x 1000 m</td>
</tr>
<tr>
<td>Simulation time</td>
<td>600 s</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>16</td>
</tr>
<tr>
<td>Number of clique CA</td>
<td>3</td>
</tr>
<tr>
<td>Number of MObile CA</td>
<td>3</td>
</tr>
</tbody>
</table>

5.1. Scenario A

5.1.1. Simulation setup

We test our protocol in the QualNet simulator Scalable-Networks (2008). In the evaluation, we compare our protocol with the manually chosen Certificate Authorities scheme (the MOCA protocol for a small network, where we set required number of CREP k to 1). Simulation parameters are shown in Table 3. All simulations are carried out in network with topology as shown in Fig. 4. In this scenario, we have not considered mobility for the sake of simplicity. Effectiveness is evaluate with the following two factors:

- The success ratio $R$ measures the effectiveness of key management protocol in request to respond. In our experiments, we...
first randomly choose a user \( N \) and a Resource, let \( N \) send request to a CA to gain admission to Resource. \( R \) is defined as follows:

\[
R = \frac{\text{Number of Request Granted}}{\text{Number of Request Sent}}
\]

- Responsive delay \( T \) measures the elapsed time in responding to request. Request should be responded within a short period, i.e., a small \( T \). \( T \) is calculated by

\[
T = \frac{(\text{Time of request granted}) - (\text{Time of sending request})}{\text{C15}}
\]

In 600-second simulation, requests are sent twice, at 400 and 500 s, respectively. We run each simulation 50 times, then get the average values of \( R \) and \( T \).

5.1.2. Simulation results

Table 4 shows the success ratio in responding to request. The proposed protocol has higher success ratio than the MOCA protocol.

It can be noticed that maximum clique nodes have better connectivity to non-clique nodes in a graph. As a result, selecting maximum clique as CAs can provide better connectivity. Besides, the choice of maximum clique as CAs is totally made by users themselves, which is suitable for the autonomous nature of MANETs.

Although maximum clique searching algorithm can return multiple results (as shown in Table 5), they have similar performances. Therefore, we simply choose the first found maximum clique as CAs.

5.2. Scenario B

5.2.1. Simulation topology

In the second scenario, we test the protocol in networks of different sizes ranging from 10 to 50 nodes. Other simulation parameters are the same as those in Scenario A.

5.2.2. Simulation results

Fig. 5 demonstrates the performance of the proposed scheme with varying size of the clique. The performance in Fig. 5 is measured by the success ratio in responding to request. The success ratio decreases as the clique size decreases, especially for a clique of small size. The reason is that there are not enough clique CAs to gather information from a large network. This results in failure to respond to request.

The time cost in finding the clique of different sizes is depicted in Fig. 6. It is evident from the figure that with a bigger clique size, the time needed to build the clique is longer. For example, in Fig. 6, the time spent in building a clique of size 3 is around 64 s while building a clique of size 7 takes much longer time (around 124 s).

Table 4

| Clique CA | 0.96 |
| MOCA CA  | 0.72 |

Table 5

<table>
<thead>
<tr>
<th>Maximum clique</th>
<th>Success ratio</th>
<th>Responsive delay (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 5, 13</td>
<td>0.96</td>
<td>7.2</td>
</tr>
<tr>
<td>1, 6, 13</td>
<td>0.959</td>
<td>6.5</td>
</tr>
<tr>
<td>1, 4, 5</td>
<td>0.9</td>
<td>6.8</td>
</tr>
</tbody>
</table>

In Fig. 7, we demonstrate the communication overhead, in terms of number of HELLO messages exchanged while building the clique of different sizes. We considered a network of 50 nodes in this scenario. As shown in Fig. 7, building a smaller clique transmits fewer number of HELLO messages (i.e., less overhead).
Fig. 8 shows a performance comparison of our proposed scheme and MOCA with randomly selected CAs. We use a network of 50 nodes. As shown in Fig. 8, the success ratio of our proposed scheme outperforms that of the MOCA protocol. This is because of the clique CAs have optimized positions rather than the randomly selected ones.

Fig. 9 presents the average delay $T$ for clique of different sizes and different numbers of nodes. The clique size ranges from 3 to 7 and the number of nodes ranges from 10 to 50 nodes. Since the average number of hops from a random node to CA in a network with bigger clique is less than that of the same network with smaller clique, the average delay is expected to be higher for cliques of smaller size. Also, because of the number of hops increases with network size, the average delay is higher.

Fig. 10 presents a comparison between our proposed scheme and MOCA for different number of CAs, with a network of 50 nodes. As shown in the figure, our proposed scheme suffers from lower delay compared with MOCA.

6. Conclusion

In this paper, we proposed a key management protocol to perform admission control for MANETs. Our proposed method offers distributed authentication service with high success ratio in respond to request, while still maintaining a low responsive delay. This admission control scheme reduces the chance of unauthorized access. Also, we demonstrate the feasibility and scalability of the proposed scheme through computer simulations.

References

Rachedi A, Benslimane A. A secure architecture for mobile ad hoc networks. In: Second international conference on mobile ad-hoc and sensor networks MSN