Binary Tree Based Public-Key Management for Mobile Ad Hoc Networks

Georgios Kambourakis, Elisavet Konstantinou and Stefanos Gritzalis

Info-Sec-Lab Laboratory of Information and Communications Systems Security

University of the Aegean, Samos GR-83200, Greece

{gkamb, ekonstantinou, sgritz}@aegean.gr

***Abstract*—The establishment of a Public Key Infrastructure (PKI) in Mobile Ad Hoc Networks (MANETs) is considered a difficult task because of the intrinsic characteristics of these networks. The absence of centralized services and the possible network partitions make traditional security solutions not straightforwardly applicable in MANETs. In this paper, we propose a public key management scheme based on a binary tree formation of the network's nodes. Using the binary tree structure, certificate chains are easily built between communicating nodes that are multi-hops away and the cumbersome problem of certificate chain discovery is avoided. We argue that our mechanism has several advantages over similar solutions, especially when a fair balancing between security and performance is terminus.**

I. INTRODUCTION

Mobile ad hoc networks (MANETs) are currently employed in many areas of interest and their self-organized nature is a challenge for researchers who wish to implement general-purpose protocols in these networks. While many routing protocols have been proposed in the literature for MANETs the establishment of a public key infrastructure (PKI) in these networks has gathered little attention so far.

The absence of any fixed infrastructure and centralized authorities makes public key management for MANETs a very difficult task from both an algorithmic and computational point of view. The main problem of any public key based security system is to make the public key of a node available to another by proving in the same time the authenticity of this key. The solution to this problem in wired, general-purpose networks comes from the use of on-line or off-line servers that issue certificates to the nodes of the network. In MANETs the absence of centralized services and the possible network partitions makes this problem very difficult. Recently, some public key management schemes for MANETs have been proposed. These schemes are classified in two categories. The first approach uses a set of nodes as servers which can provide partial certificates to a combiner by utilizing the concepts of threshold secret sharing [2]-[4]. The second category is based on the web of trust approach [5],[6]. In this approach, every node issues certificates to the nodes it trusts. By considering every issued certificate as an edge, a certification graph is formed. If two nodes wish to exchange their public keys and form a common secret, they find a certification path in the graph and they can in this way authenticate each other. However, the major disadvantage of this approach is the cumbersome problem of finding a certification path in the graph. A solution to this problem is proposed in [7] where a virtual hierarchy is built among the nodes in the graph.

In this paper we propose an approach similar to [8] which is based on a binary tree formation of the network's nodes. Specifically, two alternative methods for binary tree formation are proposed each one having its pros and cons. Using this structure, the certificate path discovery problem is avoided and the place of each node in the tree can be easily found. Moreover, the frequent join and leave events in the network are efficiently handled by modifying the tree structure where it is needed. In a nutshell, the proposed scheme has several advantages over other similar solutions, being more effective in terms of join and leave procedures, path discovery, certificate management and performance, especially when security is not of top priority. On the other hand, when security is at stake, we offer a modified version of the proposed scheme which can deliver robust security services and effectively identify Denial of Service (DoS) attacks not addressed by similar mechanisms until now. Last but not least, we discuss some methods for establishing initial trust between the nodes of a MANET. Whilst this issue is very important, as it globally affects nodes' trustworthiness, it is not adequately addressed in the literature so far. The rest of the paper is organized as follows. In Section II we show how trust can be initially established between two nodes, while in Section III we present our binary tree based protocol, consisting of two alternative tree formation mechanisms. The certificate chain discovery procedure is presented in Section IV. Section V provides a comparison with other similar methods, while last section offers concluding thoughts and some pointers to future work.

II. ESTABLISHING INITIAL TRUST BETWEEN NODES

Most works in the field of public key management assume either that some sort of trust among network entities exists beforehand or that the nodes proceed with pairwise certification blindly. After certification, if a node is detected to behave aggressively or does not obey to the network rules then its certificate is revoked or left expired. Clearly, establishing trust among network nodes in a MANET is a very challenging issue. Usually there is no external prior context at all among the participating entities. Bootstrapping from an existing infrastructure or exploiting proximity for expressing indexicality, as they are presented in [9], can furnish partial solutions towards solving this problem. For these reasons,

trust and ad-hoc networks can be thought in a sense as contradicting terms.

In many cases however, it is necessary to initialize security from the scratch for protecting subsequent interactions within the system. In this context, using proofs of work in initialization of trust, and reputation systems can assist in establishing a certain degree of trustworthiness according to the first approach also known as "proof of work" (PoW) [10],[11]. The objective behind PoW systems is that a verifier V can make sure that a prover P has successfully performed a certain computational task. The basic characteristic of a PoW system is that creating the proof must entail a predictable amount of work, while verifying the proof must be straightforward. Of course such schemes cannot fully guarantee that a node can be trusted but they assist in automatically exclude some of the bad peers from joining the network.

When no centralized authority exists, as in the case of MANET, one method towards deriving positive conclusions whether a given node can be trusted is to employ a reputation rating system [9],[12]. The reputation ratings can be based on direct experience or recommendations by others in the network community or a combination of the two. On the other hand, while reputation systems work acceptably well in centralized realms their application in MANET scenarios require a decentralized reputation system, which in turn brings several issues in the foreground mostly related with the recommendations exchange system design and the avoidance of Sybil attacks. Some other answers to the basic question "Who trust whom in a MANET and why?" do exist in terms of device authentication [13]. Yet, such solutions mandate in many cases some a priori configured trust relationship between the participating nodes. For example, every device joining the network can carry a device certificate proving its genuineness. Nevertheless, this requires a PKI infrastructure to sign all the certificates during the so-called network initialization phase. The same problem applies in the case of trusted computing fashioned solutions. In our opinion establishing trust among network entities in a MANET remains very much an open research problem.

III. Proposed Solutions

In this section we will describe two similar solutions for building a binary tree of trust between the nodes of any MANET. The binary tree approach can greatly contribute to path discovery process optimization, and thus can facilitate the acquisition of certificate chain between the involved nodes. The first one starts from a single randomly chosen node, e.g., the root of the tree and continues cascading until all willing-to-participate nodes join the tree. The other one hastens the formation of the binary tree by starting simultaneously from several different nodes.

A. The binary tree based scheme

The forming protocol starts when a given node, say N0 sends a special (extended) HELLO message (this is actually a RREP with TTL = 1) to its neighbours stating that it wants to initiate a tree-based trust relationship with them. Naturally, as there is no pre-established trust among any network nodes in a typical MANET, the adjacent nodes can accept the invitation or simply reject it. Accepting such an invitation from a given node means that the invited node is willing to proceed with a mutual-certification process with the initiator. The purpose of the protocol is to form a binary tree of trust between all network entities. So, each node can provide certificates to a maximum of two neighboring nodes. All nodes have a {public, private} key pair created locally, so for every node pair each part signs the public key of the other using its private key and sends the result towards the other part. This tree forming procedure depicted in Figure 1 continues cascading requests from the root of the tree (N0) down to the leafs. Assuming that the network is dense enough the probability of having some - willing to participate - nodes left out of this process is negligible.

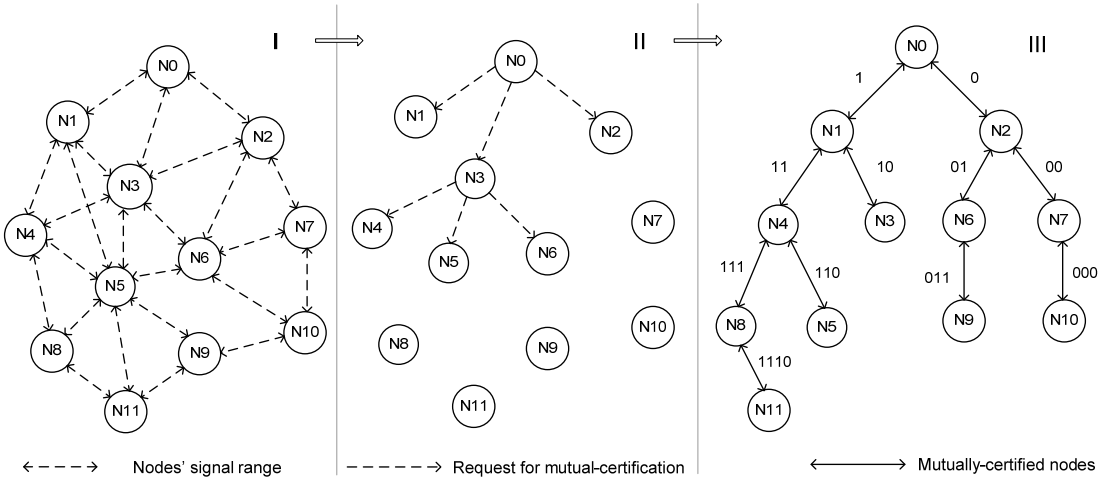


Fig. 1 Formation of the binary tree of trust

Figure 1_I depicts the initial state of the network as well as each node's signal range. At some point, N0 initiates the protocol by sending pairwise-certification requests towards N1, N2 and N3 correspondingly. The latter nodes agree to participate, so they are pairwise-certified with N0. After that, they send pairwise-certification requests towards their neighbours, e.g., N3 invites N4, N5 and N6. This situation is illustrated in Figure 1_II. The protocol continues until the binary tree depicted in Figure 1_III is formed. When a given parent-node has completed the mutual-certification procedure with two child-nodes, it will drop any similar request coming from its neighbours. For example, in Figure 1_II node N0 sends requests to N1, N2 and N3 but drops the reply from N3 since N1 and N2 have answered quicker to its request and have already been added in the binary tree. In case a child-node has already mutually-certified with a parent-node ignores post-dated pairwise certification requests send by others. To accomplish this each node must send in its HELLO messages its state in the tree, i.e., the bit 0 or 1 for non-members and members correspondingly. This is necessary in order to avoid redundant pairwise-certifications or loops between the leafs of the same tree. For instance, as N6 has set up already a relationship with N2, drops the request originating from N3. It is worth noting that all nodes are supposed to be equal and the notation "parent" or "child" denotes their position in the tree. It is also stressed that for unsecured communications the nodes can use any possible available route. For example, if N5 is in the range of N9 they can exchange data directly.

However, to establish a secure relationship they must first obtain the certificate one another via the binary tree of trust, to setup a symmetric session key, and finally communicate directly as the case may be.

As already mentioned in Section II the main question of the certification procedure remains: "how can a node be convinced that a given public key, say K(N0) truly belongs to node N0, so as to proceed with certification?" Whilst all the aforementioned solutions can be applied in our case, we adopt a "commitment-driven" solution. That is, every node commits itself to the scheme; to be disciplinarian and behave legitimately. Therefore, initially, every node certifies the other for a sort period of time, say for some hours. After that, if the aspirant node proves good intentions its certificate is renewed with a greater validity period. It is worth noting that detecting misbehaving nodes among one-hop nodes is quite easy due to the broadcast nature of wireless communications. The certified node must present a valid certificate to get a new one. Otherwise, the renewal procedure fails. Even though the proposed method imposes increased node overhead during its first stages, balances some time later after achieving a relative high degree of trust level between all participants.

B. The parallelized binary tree based scheme

The binary tree based scheme described in the previous section, can be easily parallelized in order to improve efficiency. Instead of starting the protocol with a given node, one can initiate the protocol by using two or more nodes. The number of these nodes can be a parameter in the whole network. Every such root node leads to the construction of a small binary tree (which can be considered as a small cluster) and all these trees can be linked together by their root nodes forming a bigger network of trust. Linking different binary trees into one also implies that every node on each tree carries also the unique identity of the tree, i.e., the IP address of the root.

Consider for example, the network in Figure 2_I. Suppose that nodes N0, N4 and N11 are randomly selected and they start the execution of the binary tree based scheme. After, the first step of the protocol, three subtrees have been created (see Figure 2_II). Every subtree should have a unique tree_ID e.g., the IP address of the root node. When a node accepts an invitation from one of its neighbours, it should check whether this node has the same tree_ID. If the two nodes' tree_IDs are the same, then the invited node does not accept the invitation (otherwise a cycle would be formed). In the case that the tree_IDs are different, then both nodes agree randomly in one of the tree_IDs and inform all the other nodes in the two subtrees in order to all adopt the same tree_ID. For example, in Figure 2_III node N8 has sent an invitation to node N11, node N11 has accepted it (since N8 and N11 belong to different subtrees) and the rest of the nodes are notified that they belong now in the same binary tree. If N8 sends an invitation now to its other neighbour (N5), then this request will be denied since N8 and N5 belong now in the same subtree.

However, there is another one parameter that should be taken care of in order to guarantee that a binary tree is created. A node having accepted three invitations should not accept another one, even in the case that this request is coming from a different subtree node. If this restriction is not satisfied, then the formed tree would not be binary. When all nodes in the network have been visited (Figure 2_III), a node having two adjacent edges should be chosen to be the root. For example, if node N4 is chosen in Figure 2_III then the formed tree is the one in Figure 2_IV. Generally, this scheme performs faster when compared to those described in the previous subsection. However, this comes at a cost in complexity, i.e., the merging process of different subtrees.

C. Join and leave events

According to the proposed schemes the join and leave procedures are straightforward. Briefly, when a node leaves or an entrant joins the network only a branch of the tree is affected. More specifically, supposing that N4 in Figure 1_III leaves the community, e.g., moves out of range, nodes N8, N5 will seek parent in N3 or N6 depending on the topology and signal strength.

On the other hand, thinking of a scenario where N12 joins the network near the range of N3 it will establish a relationship with either N3 or N5, N6, N9. It is implied that in the rare case a newcomer cannot immediately find an association it must wait for some time until some other node roams out of that specific coverage area (a parent loses one child). In such occasions there is always the possibility for the node to roam to a new position until it finds a pair.

Lastly, the most complex leave situation is when the root node, say N0 in Figure 1_III, leaves the tree for some reason. Then N1 or N2, that is, the nodes closer to him, must replace N0. Assuming that N1 takes over the role of the root he must abandon N3, keep the connection with N4, and establish a direct relationship with N2. Consequently, N3 must seek for another parent. Even in this case the join procedure is expected to complete after very few interactions, i.e., new mutual-certifications between the corresponding nodes.

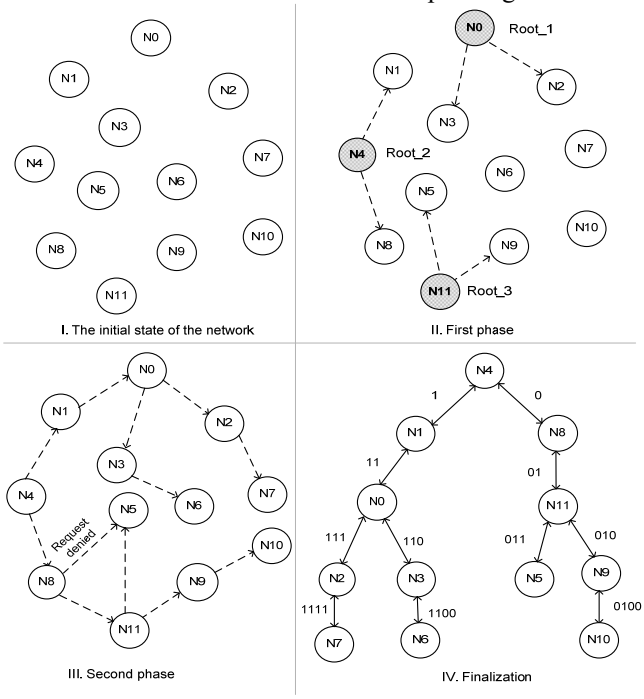


Fig. 2 Example of the parallelized binary tree based scheme

IV. CERTIFICATE CHAIN DISCOVERY PROCEDURE

For secure communication any two nodes must be authenticated mutually. This means that each part must acquire and verify the certificate of the other. This can be fulfilled by constructing a certificate chain between them. In the following we consider an approach based on Ad hoc On-Demand Distance Vector (AODV) [1]. However, our method can be embedded through proper extensions or slight modifications to any existing routing mechanism like Dynamic Source Routing (DSR), Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) and Cluster Based Routing Protocol (CBRP) to mention just a few.

AODV defines three message types which are Route Requests (RREQs), Route Replies (RREPs), and Route Errors (RERRs). All message types are received via UDP, and normal IP header processing applies. According to AODV, every time a route to a new destination is needed, the node broadcasts a RREQ to discover a route to the destination. Note, that a route can be determined either when the RREQ reaches the destination itself, or an intermediate node that holds a fresh route to the destination [1]. Upon that, the route is made available to the initiator of the RREQ by unicasting a RREP back to him. This is possible because each node receiving the initial request caches a route back to the originator. The binary structure further assists route discovery as each branch of the tree can be quickly identified by a binary sequence. For instance, referring to Figure 1_III and starting always from the root, the route to N5 is '110', where '1' means left and '0' right. This fact actually revokes the need for route optimization in every hop making the whole procedure particularly effective. Taking Figure 1_III for example, in the following we describe the necessary steps for N8 to build a certification chain with N10.

(a) To set up the required certificate chain N8 broadcasts a RREQ towards N10. This means that the IP address of N10 is already known (maybe from a previous RREQ). In order to indicate to the destination that this RREQ aims to a certificate chain establishment, N8 inserts the value '11' in the RREQ reserved field as shown in Figure 3_I. Note that this field is always sent as '0' in AODV and is ignored on reception [1]. By setting the '11' value in the reserved field, N8 also ensures that this specific RREQ will reach its final destination. That is, every intermediate node must forward the RREQ to its final destination.

(b) Upon reception of RREQ N10 constructs a corresponding RREP. First, it marks the reserved field with the value '11' meaning that the packet refers to a certificate chain reply. Finally, N10 appends its own certificate to the message, signs the {RREP || Cert(N10)} block using its private key and appends it to the RREP. The format of the modified RREP packet is depicted in Figure 3_II. The resultant packet is sent back to N8 as a reply.

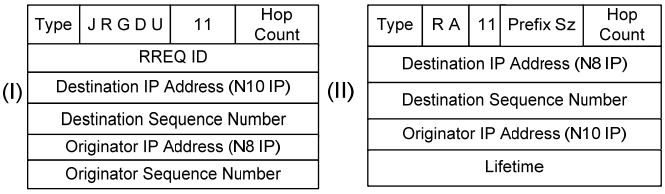


Fig. 3 RREQ towards N10 & RREP towards N8

(c) All intermediate nodes must inspect every RREP that contains '11' in the reserved field. Specifically, N7 will check the genuineness of the signature Sig(RREP || Cert(N10))N10 contained in the received message. Note, that N7 has the public key of N10, so it can securely verify the signature. Every attempt by means of a man-in-the-middle attack to alter the certificate of N10 or the original RREP will produce an error. Assuming that the signature check returns true, N7 will sign the {RREP || Cert(N10)} using his own private key and forward the result along with the {RREP || Cert(N10)} block to N2. All the nodes in the path, that is, N2, N0, N1, N4 will repeat the same steps, as N7 did, until RREP reaches N8. If an error occurs at a given hop, that is the signature is not valid, the process is halted, and a RERRs is generated and forwarded back to the initiator. We should mention that it is important for every node in the chain to sign not only the original RREP but also the

certificate of the target node (N10 in our case). This is done to prevent DoS attacks where the certificate of say N10 is altered when in transit by a man-in-the–middle attacker. So, N8 will receive a bogus certificate of N10. Later on, N8 and N10 having the certificates one another will try to establish a symmetric key by utilizing e.g., a challenge – response protocol. Naturally, the process will fail because the public key of N10 is not valid. This of course leads to a DoS and consequently, the nodes must repeat the certificate chain discovery procedure from the beginning.

(d) As soon as {RREP || Cert(N10) || Sig(RREP || Cert(N10)N4 } arrives to its final destination, N8 will check the validity of the signature using the local copy of N4's public key. If everything is correct N8 will prepare his RREP to be sent towards N10. The procedure is exactly the same as before but in the reverse order. This will allow N10 to successfully acquire the certificate of N8. Figure 4 provides an overview of all the aforementioned steps.

(e) After the two ends have acquired the certificate of each other they can agree on a per session shared secret (symmetric key) to communicate securely.

V. Analysis & Comparison with related work

Several mechanisms for self-organized public key management in MANETs have been proposed in the literature so far. All solutions can be briefly categorized into localized schemes [6],[14], cluster based [8] and hierarchical ones [7]. Naturally, our solution is also classified into the hierarchical category. According to localized schemes each node must be mutually-certified with all its neighbours. As a result, the overhead for issuing, storing and maintaining certificates is far larger when compared to our method. Moreover, join and leave procedures for localized mechanisms are generally more complex and require frequent interplay between many nodes of the network. According to our solution each node is associated with maximum three other nodes and the join and leave procedures are straightforward, excluding that of which the root node leaves the tree (see section III.C).

Another important issue is that localized schemes build one-way trust relationship, not mutual. Putting it another way, only the certificate of the destination is acquired by the initiator, not the opposite as our scheme mandates. However, in order for the two entities to become mutually authenticated each one of them must successfully obtain the public key certificate of the other. Even worse, according to localized schemes, the certificates of all the involved in the chain nodes are stacked all the way back to the initiator. Therefore, the more nodes in the certification path the bigger the RREP message towards the initiator will be. On the downside, our mechanism copes with this problem similarly to [8] which is a cluster based protocol. Specifically, each node in the path must locally authenticate any certification path message received from a previous node in the path. If this check fails then an error message is instantly sent towards the initiator. Furthermore, according to [8] each node in the path must send its certificate to the next mode for validating the RREP's signature. However, this is not necessary (on the contrary it creates extra overhead to each node) because every node retains a pairwise-certification relationship with the previous and next one in the path. So, every node uses its local copy of the certificate of the previous node to validate the signature. Actually, this is the safest way to do this. To the best of our knowledge, localized schemes do not protect the integrity of the vital parts of the certificate chain messages by having each node signing them, thus they are prone to DoS attack scenarios. Another issue with cluster based oriented solutions like [8] is that only the RREP is integrity protected (signed); not the certificate of the destination node. This of course can lead to DoS attacks as already described in Section III. Moreover, the certificate chain discovery procedure in [8], which as already mentioned is cluster based, requires route optimization in each hop, which is also avoided by our scheme; actually all routes are already optimized due to the (virtual) binary-tree-style topology.

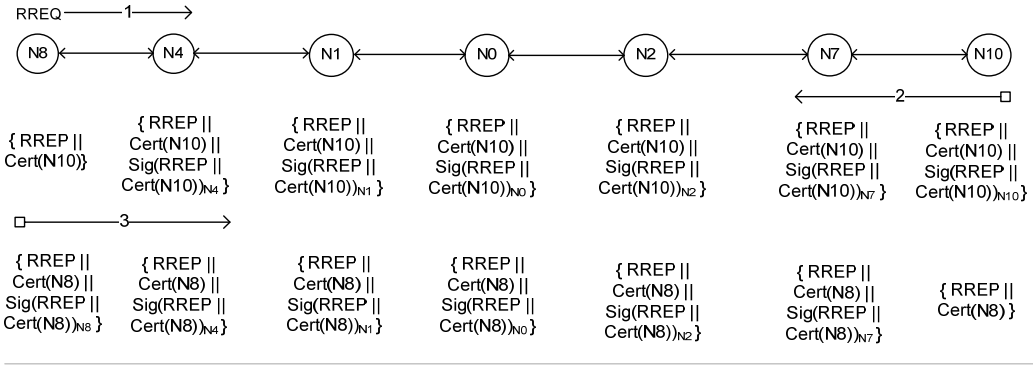
One possible problem with all solutions arises when a node in the path becomes compromised. In such an event, the malicious node could craft the certificate of the destination, construct as normal the: $\text{Sig(RREP || bogus\_Cert(Destination))}_{\text{private\_key\_of\_malicious\_node\_y}}$ part and forward the message to the next hop as usual. The next node in the chain is not able to detect the forgery, so the initiator will eventually receive a bad certificate. This situation also leads to a DoS incident and breaks down the whole chain. Note, that no solution proposed so far deals with such an attack. Actually, alike scenarios can be avoided with a significant extra overhead. More analytically, each node in the path, after putting its own certificate, must re-sign the Sig(RREP || Cert(initiator))y part over again with its private key and append each own certificate to the message too.

For instance, the corresponding message for N7 to be sent towards N2 will be: {RREP || Cert(N10) || Cert(N7) || Sig(Sig(RREP || Cert(N10))N10 || Cert(N7))N7}. By doing so, the initiator (N8) will finally receive a re-encapsulated signature by all nodes in the path as well as all nodes' certificates. Starting from the inner signature, N8 can sequentially recalculate all the signatures until the outer one. Actually, N8 can detect the point of failure and alert other nodes to exclude the misbehaving member. Although this procedure adds more overhead, has also a positive outcome. That is, each node acquires valid certificates of all other nodes in the chain. Storing the certificates until they become expired can accelerate future communications. In every case one must balance wisely between performance and security. So, if security is terminus then the aforementioned solution must be followed.

VI. Conclusions and future work

In this paper we proposed a public key management scheme based on a binary tree formation of the network's nodes. Specifically, we discussed and analyzed two variations of a method for building a binary tree of trust between the nodes of MANET. The first one initiates the formation

procedure from a single randomly chosen node, while the other hastens the formation of the binary tree by starting simultaneously from several different nodes. As future work we would like to thoroughly evaluate our mechanism in terms of service times and security robustness.



Where $Sig(x)_y$ is the signature of block x signed by the private key of node y and Cert(x) is the public key certificate of node x.

Fig. 4 Certificate chain discovery procedure between N8 and N10

REFERENCES

[1] Perkins, C., Belding-Royer, E., Das, S., "Ad hoc On-Demand Distance Vector (AODV) Routing", IETF RFC 3561, July 2003.

[2] Zhou, L., and Haas, Z. L., "Securing ad hoc networks", IEEE Network, Vol. 13, Issue 6, pp. 24-30, IEEE press, 1999.

[3] Kong, J., Zerfos, P., Luo, H., Lu, S., Zhang, L., "Providing robust and ubiquitous security support for mobile ad-hoc networks", Proceedings of ICNP 2001, pp. 251-260, Nov. 2001.

[4] Yi, S., Kravets, R., "MOCA: Mobile certificate authority for wireless ad-hoc networks", Proceedings of PKI '03, 2003.

[5] Hubaux, J.-P., Buttyan, L., Capkun, S., "The quest for security in mobile ad hoc networks", proceedings of ACM Mobihoc, 2001.

[6] Li, R., Li., J., Liu, P., Chen, H.-H., "On-demand public-key management for mobile ad hoc networks", Wireless Communications & Mobile Computing, Vol. 6(3), pp. 295 - 306, May 2006, Wiley.

[7] Satizábal, C. Hernández-Serrano, J., Forné, J., Pegueroles, J., "Building a virtual hierarchy to simplify certification path discovery in mobile ad-hoc networks", Computer Communications, Vol. 30 , Issue 7, pp. 1498-1512, May 2007, Elsevier.

[8] Hahn, G., Kwon, T., Kim, S. and Song, J., "Cluster-Based Certificate Chain for Mobile Ad Hoc Networks", proceedings of the ICCSA 2006, pp. 769-778, LNCS 3981, Springer.

[9] Asokan, N., Tarkkala, L., "Issues in initializing security", proceedings of ISSPIT, pp. 460 – 465, 2005.

[10] Dwork, C., Naor, M., "Pricing via Processing or Combating Junk Mail", CRYPTO'92, pp. 139-147, 1992.

[11] Abadi, M., Burrows, M., Manasse, M., Wobber, T., "Moderately Hard, Memory-Bound Functions", ACM Trans. Inter. Tech., Vol. 5, No. 2, pp. 299-327, 2005.

[12] Marias, G. F., Flitzanis, D., Mandalas, K., and Georgiadis, P., "Cooperation Enforcement Schemes for MANETs: A Survey", Wiley's Journal of Wireless Communications and Mobile Computing, Special Issue, Vol.6, Issue 3, pp. 319-332, 2006.

[13] Kambourakis, G., Gritzalis, S., "On Device Authentication in Wireless Networks: Present issues and future challenges", TrustBus'07, LNCS, 2007.

[14] Capkun, S.; Buttyan, L., Hubaux, J.-P., "Self-organized public-key management for mobile ad hoc networks", IEEE Transactions on Mobile Computing Volume 2, Issue 1, Jan.-March 2003, pp. 52 – 64, IEEE press.