MOCA : Mobile Certificate Authority for Wireless Ad Hoc Networks

Seung Yi, Robin Kravets
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
{seungyi,rhk}@cs.uiuc.edu

Keywords : PKI, Key Management, Security, MANET, Ad Hoc Networks, Threshold Cryptography

Abstract

PKI has been recognized as one of the most effective tools for providing security for dynamic networks. However, providing such an infrastructure in ad hoc wireless networks is a challenging task due to their infrastructure-less nature. In this paper, we present these challenges in detail, identify the requirements for such solutions, and propose a practical PKI service for ad hoc networks. We employ threshold cryptography to distribute the CA functionality over specially selected nodes based on the security and the physical characteristics of nodes. The selected nodes that collectively provide PKI functionality are called MOCA (MObile Certificate Authority)s. Using these MOCAs, we present an efficient and effective communication protocol for correspondence with MOCAs for certification services. Results from our simulations verify the effectiveness and the efficiency of our approach.

1 Introduction

Since its birth more than two decades ago [20], public key cryptography has been recognized as one of the most effective mechanisms for providing fundamental security services including authentication, digital signatures and encryption. The effective management of digital certificates is a key factor for the successful wide-spread deployment of public key cryptography. PKI (Public Key Infrastructure), an infrastructure for managing digital certificates, was introduced exactly for this purpose [19]. The most important component of PKI is the CA (Certificate Authority), the trusted entity in the system that vouches for the validity of digital certificates. The success of PKI depends on the security and availability of the CA to the principals in a system (or the nodes in a network) since a principal must be able to correspond with the CA to get a certificate, check the status of another principal's certificate, acquire another principal's certificate, and so on. PKI has been deployed for wired networks [3, 1] and some infrastructure-based wireless networks [9]. Since good connectivity can be assumed in these networks, the main thrust of research in such environments has focused on the security of the CA and the scalability of the CA to handle a large number of requests.

However, it is unclear if such approaches can be extended to ad hoc networks. A wireless ad hoc network or a mobile ad hoc network (MANET) [13] is a network where a set of mobile devices communicate among themselves using wireless transmission without the support of fixed or stationary infrastructure. Due to its infrastructure-less nature, an ad hoc network can be deployed very fast at a relatively low cost enabling communication when it is not possible or too expensive to deploy a support infrastructure. A wide range of military and commercial applications have been proposed for ad hoc networks. For example, a unit of soldiers moving in the battlefield cannot afford to set up a base station every time they proceed to a new area. Similarly, setting up a communication infrastructure for a casual and spontaneous conference meeting among a small number of people cannot be justified financially. Additionally, ad hoc networks can be the perfect tool for a disaster recovery or emergency situation when the existing communication infrastructure is either destroyed or disabled. A large portion of research in ad hoc networks has focused on routing, medium access control and power management and only recently researchers have started looking at security issues in ad hoc networks.

Connectivity, which was assumed to be good in previous PKI solutions, is not easy to maintain in ad hoc networks. On the contrary, maintaining connectivity is one of the main challenges, since the inherent infrastructure-less nature of ad hoc networks inhibits guaranteeing any kind of connectivity. Another serious problem present in ad hoc networks is the increased physical vulnerability of the nodes themselves. Considering that many ad hoc networks will be deployed with mobile nodes [12], the possibility of the nodes being captured or compromised in a hostile environment is higher than in wired networks with stationary hosts. Mobile nodes in infrastructure-based wireless networks have the same vulnerability, but they can rely on the infrastructure for detection of compromised nodes, help with recovery and storage of sensitive information. Since there is no stable entity in an ad hoc network, ad hoc nodes cannot enjoy such conveniences.

Several proposed solutions for providing PKI for ad hoc networks address the increased vulnerability of the mobile nodes by employing techniques to distribute the CA functionality across multiple nodes, generally using threshold cryptography [21, 15]. These approaches also increase the availability of the CA. While these approaches share some similarities with the MOCA framework, they are either conceptual [21], not targeted for ad hoc networks [22], or vulnerable against attacks [15]. The MOCA framework provides a practical and secure key management framework for ad hoc networks with communication support that considers the dynamic nature of connectivity in ad hoc communication.

We identify two main challenges in distributing the CA functionality over multiple nodes. The first challenge is picking a set of nodes to collectively provide the CA service. The second and equally important challenge is how to provide efficient and effective communication between the mobile nodes and the CA nodes, even in dynamic networks with possible compromises or temporary network partitions.

To this end, we present the MOCA (MObile Certificate Authority) framework. A MOCA is a mobile node within an ad hoc network selected to provide distributed CA functionality. A network operator chooses MOCAs based on an observation of heterogeneity among mobile nodes, typically physically more secure, computationally more powerful, or more trustworthy nodes. MOCA nodes use threshold cryptography to share the responsibility and provide CA services with strong security and high availability. Client nodes are equipped with MP (MOCA certification Protocol) that enables contacting sufficient MOCAs in an efficient and effective way. We demonstrate the effectiveness of our protocol with extensive simulations. Based on simulation results, we also provide certain insights into how to configure such security services for ad hoc networks.

The remainder of this paper is organized as follows. In Section 2, we define important metrics for designing key management frameworks for ad hoc networks and use these metrics to evaluate some existing research. In Section 3 and 4, our approach using MOCAs is presented and Section 5 presents simulation results from our implementation. Section 6 suggests some possible extensions of this work and we conclude in Section 7.

2 Key Management for Ad Hoc Networks

Any successful key management framework for ad hoc networks requires fault tolerance, security, and availability. These terms are sometimes used interchangeably, mainly because they are not independent of each other. To avoid confusion, we clearly define these terms and apply them to some existing approaches for evaluation.

Fault Tolerance: The main concern of fault tolerance is the capability to maintain correct operation in the presence of faulty nodes. We restrict the definition of faulty nodes to observable faults. If a node is malfunctioning and other nodes can observe such malfunctions, a certain level of recovery is possible. We employ intelligent replication using threshold cryptography to provide tolerance of faulty nodes.

Security: Acting as the trust anchor for the whole network, the MOCA framework should be secure against malicious nodes or adversaries. While it may not be possible to be resistant to all levels of attacks, there should be a clear threshold of attacks a system can withstand while operating normally. MOCA nodes are selected based on their node characteristics and they form a distributed CA to resist adversaries.

Availability: Traditionally, the term availability has been used in conjunction with fault tolerance. But in ad hoc networks availability is also highly dependent on the connectivity of the network. In wired networks, if there are no faulty or compromised nodes, the system is by definition available for clients since connectivity is not a problem. In ad hoc networks, even when there are no faulty or compromised nodes, clients may not be

able to contact the desired services due to inconsistent connectivity. We provide a set of efficient and effective communication protocols for clients to contact MOCAs to address availability.

The simplest approach to providing CA functionality in an ad hoc network is to assign a single node to be the CA. The success of this scheme depends on that single CA node. This approach is not fault tolerant, since failure of one node breaks the system. Similarly this approach is highly vulnerable, since an adversary need only compromise one node to acquire the secret key. Finally, given the expected mobility and unpredictability of ad hoc networks, it may be possible that nodes will not be able to reach the CA in a timely fashion, making availability very unpredictable. Therefore, a single CA cannot effectively service a whole ad hoc network.

Robustness, which is missing in the single CA scheme, can be achieved by replicating a fully functional CA on *r* different nodes. With *r* replicas, the system can withstand *(r - 1)* failures because the CA service is available as long as there is at least one operational CA. Availability has also been improved since a client node has a better chance of reaching one of *r* CAs to get service. Unfortunately, the system has become more vulnerable. An adversary need only compromise one of the *r* CA nodes to acquire the secret key and so compromise the whole system. Therefore, using replicated CAs is not a viable solution in ad hoc networks. The problem of using replicated CAs stems from the fact that each replica has full knowledge of the system secret.

Zhou and Haas first proposed to use threshold cryptography to securely distribute the CA's private key over multiple nodes to form a collective CA service [21]. Using *k* out of *n* secret sharing [4] can provide a good level of fault tolerance and security. They address the problem clearly and present conceptual design issues, but do not address the problem of availability in their work. The authors continued their work in COCA, which is a distributed CA approach using threshold cryptography [22], designed to serve networks like the Internet. Again, connectivity between clients and the distributed CAs was not a concern.

Kong and others address availability by making all *M* nodes in the network share CA functionality [15]. A client need only contact *k* out of *M* nodes to get a certification service. Assuming there are *k* nodes in a client's one hop neighborhood, the client can get a certification service cheaply by using a one-hop broadcast for the request. While this solution addresses availability and fault tolerance, it compromises the security of the system. In general, the gap between *k* and *n* in secret sharing schemes defines the security of the system. *k* can be chosen between 1 and *n* in any secret sharing scheme. As *k* approaches *n*, thus closing the gap between *k* and *n*, the system becomes more secure because an adversary needs to compromise at least *k* nodes to collapse the system. But if *k* is too large, the system becomes less available to clients and also less tolerant to faults. When *k* approaches 1, making the gap larger, the effect is reversed and the system becomes more available but also less secure. Kong chose to keep *k* relatively small to address the availability problem and ended up with a vulnerable system where any adversary need only compromise a small number of nodes in the network to collapse the service.

Another notable scheme is proposed by Hubaux and others [10]. In their scheme, there is no concept of a CA. Every node acts as its own CA, similar to the PGP "Web of Trust" model. The main difference between PGP and their scheme is that there is no longer a well-known certificate directory where all certificates are stored. Rather, every user in the system carries a part of the certificate directory. In PGP, when two users wish to authenticate each other, they must search the certificate directory for a chain of certificates that links both users. In Hubuax's scheme, this problem is transformed into finding an intersecting point between the certificate chains carried by each user. Hubaux proposed a shortcut-hunter algorithm for this problem. While their approach is practical for the totally self organizing networks they aimed at, it has the inherent problem of no definite trust anchor like the CA in other CA-based PKI approaches. Therefore, it is not meaningful to evaluate this scheme using our framework.

3 MOCA

In this section, we present a practical key management framework for ad hoc networks. We first discuss the impact of heterogeneity among mobile nodes in a given ad hoc network and how it can be exploited to help choosing the MOCA nodes better. Then, we describe the details of MOCA framework and present a set of parameters to tune our framework.

3.1 Heterogeneity within an Ad Hoc Network

Most research in ad hoc networking has implicitly treated all nodes as identical in many respects, including power, transmission range, computational capacity, and security. We contend that mobile nodes in many ad hoc networks will be heterogeneous in many respects, especially in terms of their security and that any security service or framework should utilize this *environmental* information. For example, consider a battlefield scenario with a military unit consisting of infantry soldiers, platoon commanders' jeeps, company commanders' command vehicles, artillery vehicles, transport vehicles, and even tanks. All of these nodes may have different ranks, power, capabilities, transmission ranges, levels of physical security, and so on. In such a case, it would be wise to pick nodes with higher ranks, more power, more capabilities to provide any security service to the rest of the network. While it may not be necessary to exploit this potential heterogeneity to enhance the basic ad hoc routing itself, certainly this heterogeneity can be used to make the network more secure by endowing "better" nodes with more sensitive information.

Similar situations can be imagined in emergency rescue operations, mining operations, or any other scenarios where ad hoc networks can play a critical role to facilitate operations. Even in a simple school field trip, it makes more sense to allow teachers to perform sensitive operations instead of students. In general, knowledge of such heterogeneity should be used to determine the nodes that will share the responsibility of the CA. For example, the chosen nodes could be the most physically secure nodes with maximum resources that are least prone to compromises or failure.

It may seem counterintuitive to limit the candidate nodes for MOCAs to a subset of mobile nodes. This decision puts a limit on the maximum number of MOCAs in the system, which in turn may reduce the level of security and fault tolerance achieved by the distributed nature of MOCAs. For example, in a 300 node network, an operator may have the choice of selecting 200 random nodes to support CA functionality. Or the operator may pick 30 nodes with higher physical security. Blindly comparing the number of MOCAs in the system, the first approach looks better because it has more MOCAs in the network, improving fault tolerance and availability. But by guaranteeing an adequate level of security of the 30 MOCAs in the second case, compromising them can be made much harder than compromising the randomly selected MOCAs in the first case, hence making the second case more secure against adversaries.

It is possible that an ad hoc network does not have enough heterogeneity among the nodes, which may make it difficult if not impossible to choose MOCAs based on this heterogeneity assumption. In such cases, we can fall back to random sampling to choose MOCAs. Our protocol still works as designed but the level of security will decrease since there is no guarantee on the security of each MOCA.

3.2 MOCA Framework

In our framework, n MOCA nodes provide the functionality of a CA to the whole network. Using threshold cryptography, these n MOCAs share the CA's private key and any set of k MOCAs can reconstruct the full CA key.

Threshold cryptography is an application of secret sharing that was first proposed by Shamir [4]. The basic idea of secret sharing is that it is mathematically possible to divide up a secret to n pieces in such way that anybody who requires the full secret can collect any k pieces out of those n to reconstruct the full secret. k becomes the threshold needed to reconstruct the secret. Threshold cryptography applies this technique to the keys for the cryptographic operations. Frankel and Desmedt [8] proposed to use secret sharing for the private key of public key cryptography and Shoup proposed a way to generate a digital signature from key pieces without reconstructing the full key at any point [18].

With a naive implementation, the CA's private key gets reconstructed per request at the client. To prevent this, we use threshold digital signatures [18]. Any client requiring a certification service must contact at least k MOCAs with its request. The contacted MOCAs each generate a partial signature over the received data and send it instead of sending their key share. The client needs to collect at least k such partial signatures to reconstruct the full signature and successfully receive the certification service.

Maintaining information on revoked certificates is one of the key tasks of the CA and this topic has received much attention in recent years [5]. In the MOCA framework, we use the simple certificate revocation list (CRL) approach and we plan to investigate a more adequate means of certificate revocation in ad hoc networks in the future. In the current framework, again k or more MOCAs must agree to revoke a certificate. Each MOCA generates a *revocation certificate* that contains which certificate to revoke and signs it with its key share. Then, each MOCA broadcasts the partially signed revocation certificate. Any node that collects k or more such partially signed revocation certificates can reconstruct the full revocation certificate. The list of revoked certificates or the CRL can be maintained by any node in the network since revocation certificates are not secrets but public information. The CRL can be stored at each

node, the MOCAs, or at a set of specially designated nodes. To avoid false revocation, unless the MOCA framework is compromised, it is not possible to forge a revocation certificate with a valid signature on it. In the MOCA framework, the partial revocation certificates are distributed to all nodes in the ad hoc network via a network-wide flood. While this imposes significant overhead on the network, we would expect a revocation to be a rather infrequent event and the cost would be amortized over time. In our current work, we are considering techniques to provide more efficient support for revocation.

3.3 Tuning Threshold Cryptography

The shape of a MOCA framework is determined by the total number of nodes in the network, the number of MOCAs, and the threshold value for secret reconstruction. Although the total number of nodes in the network, *M*, can change dynamically over time, it is not a tunable parameter. The number of MOCAs, *n*, is determined by the characteristics of nodes in the network, such as physical security or processing capability and it is also not tunable. In this system, *n* defines the limits of the system as an upper bound for *k*, the minimum number of MOCAs a client must contact to get certification services.

Given *M* and *n*, the last parameter *k*, the threshold for secret recovery, is indeed a tunable parameter. Once *k* has been chosen and the system is deployed, it is expensive to change *k*. Therefore it is important to understand the effects of varying *k* on a given system.

k can be chosen between 1 (a single CA for the whole network) and *n* (a client needs to contact all MOCAs in the system to get certification services). Setting *k* to a higher value has the effect of making the system more secure against possible adversaries since *k* is the number of MOCAs an adversary needs to compromise to collapse the system. But at the same time, a higher *k* value can cause more communication overhead for clients since any client needs to contact at least *k* MOCAs to get certification services. Therefore, the threshold *k* should be chosen to balance the two conflicting requirements. It is clear that no one value will fit all systems. Our goal is to provide some guidelines for choosing an appropriate *k*. To make our protocol more adaptive to varying network configurations, we introduce additional tunable parameters in Section 5.

4 MOCA Certification Protocol

In this section, we describe a key aspect for successful PKI in ad hoc networks: *communication*. The choice of which and how many MOCAs to contact must be made in coordination with the communication protocol used to access the MOCAs. Even after MOCAs have been selected and deployed in the system, it is useless if clients cannot contact them and receive services. The communication pattern between a client and *k* or more MOCA servers is *one-to-many-to-one*[1], which means that a client needs to contact at least *k* MOCAs and receive at least *k* replies. To provide an effective and efficient way of achieving this goal, we propose MP (MOCA certification Protocol).

In MP, a client that requires certification services sends Certification Request (CREQ) packets. Any MOCA that receives a CREQ responds with a Certification Reply (CREP) packet containing its partial signature. The client waits a fixed period of time for *k* such CREPs. When the client collects *k* valid CREPs, the client can reconstruct the full signature and the certification request succeeds. If too few CREPs are received, the client's CREQ timer expires and the certification request fails. On failure, the client can retry or proceed without the certification service.

The CREQ and CREP messages are similar to Route Request (RREQ) and Route Reply (RREP) messages in on-demand ad hoc routing protocols like AODV [7] and DSR [11]. The management of routing information is also similar to these protocols. As a CREQ packet passes through a node, a reverse path to the sender is established. These reverse paths are coupled with timers and maintained long enough for a returning CREP packet to be able to travel back to the sender. If no CREP is returned within the time-out period, the reverse path entry in the routing table expires and is purged. If a CREP traverses back through the previously set-up reverse path to the sender, the routing table entries are refreshed and the bidirectional path remains in the routing table for potential reuse. This similarity to on-demand routing presents a potential for our certification protocol and the existing on-demand routing protocols to benefit from each other by sharing routing information.

[1] We term this pattern of communication "Manycast".

4.1 Flooding

The simplest means of reliable data dissemination, flooding, can be used to reach all MOCAs in the network [17]. As shown in previous results, while this flooding approach is effective, it generates a large amount of traffic. First, the overhead generated from a network-wide CREQ flood is large. Second, since a client has no way to limit the dissemination of a CREQ, all the MOCAs that receive a copy of the CREQ respond with a CREP and the client receives more responses than it actually needs to reconstruct the full signature. Any partial signatures beyond the required *k* are discarded and waste networking and processing resources.

4.2 Unicast-based Optimization

To reduce the amount of overhead from flooding while maintaining an acceptable level of service, we introduce β-unicast, where the client can use multiple unicast connections to replace flooding if the client has *sufficient* routes to MOCAs in its routing cache. β in the name represents the *sufficient* number of cached routes to MOCAs to use unicast instead of flooding. If this sufficiency is achieved, β-unicast sends multiple unicast CREQs instead of flooding the network with CREQs. β-unicast does not initiate any form of route discovery as in on-demand ad hoc routing protocols where a network is usually flooded with route discovery packets. Instead, β-unicast only utilizes the existing information in the route cache. Blind use of unicast with insufficient cached routes can result in service failure, which in turn causes another round of flooding. To prevent such a situation, our protocol uses flooding when there are not enough routes cached.

The definition of sufficiency is tightly coupled to the value of *k*, but is also highly dependent on the state of the network. If the network is very stable with low mobility, having just *k* cached routes may be sufficient since the client can expect to receive all *k* replies back. If the the network is highly mobile and routes are unstable, sending out exactly *k* unicast CREQs is dangerous since even one loss of a CREQ or a CREP results in the failure of the whole certification request. In this situation, the node should send out additional CREQs to increase the probability of success. The number of additional CREQs is defined by α, a marginal safety value used to increase the success ratio of β-unicast. α is node specific and can be determined based on the node's perception of the network status. How a node will perceive the status of the network is out of the scope of this paper and is an active topic of research. The sum of the crypto threshold *k* and the safety margin α is the unicast threshold, β, hence the name β-unicast.

Our previous work showed that a client often has a moderate number of cached routes to MOCAs under reasonable certification traffic in the network [17]. A result given in [17] shows that under a mobility of 10 second pause time and 10 $\mathrm{m/s}$ maximum speed, clients have cached routes to 45% of the MOCAs on average. One interesting question is how to choose among the MOCAs cached in the routing table. If there are only β cached routes, the client needs to contact every one. But if there are more than β routes in the cache, the choice of which ones to use can affect performance. We define three different schemes:

1. Random MOCAs - Choose β random MOCAs with cached routes.
2. Closest MOCAs - Choose β MOCAs with smallest hop counts in the cache. Intuitively, this approach has the benefit of the shortest response time and the smallest packet overhead since the CREQ packets travel the least distance.
3. Freshest MOCAs - Choose β MOCAs with the freshest cache entries. The most recently added or updated entries should not be stale, especially under high mobility. By choosing the freshest MOCAs, the client should be able to minimize the risk of failure under high mobility.

We provide simulation results for these certification protocols in the next section.

5 Evaluation

The focus of our evaluation of the MOCA framework is effectiveness and efficiency (or cost). Effectiveness is measured using the success ratio of certification requests. For flooding based protocols, success ratio is defined as the total number of received CREPs. For unicast-based optimizations, every CREQ that receives *k* or more CREPs is counted

as a successful certification request and success ratio is defined as:

$$\frac{(Number\ of\ successful\ certification\ request)}{(Number\ of\ total\ certification\ request)}$$

The cost of a certification protocol can be evaluated using the two metrics: packet overhead and additional communication delay caused by the certification process. The simulations demonstrate that our approach is practical for ad hoc networks providing adequate service availability without incurring prohibitive overhead.

For all simulations, there are three parameters that can be tuned according to the network configuration.

- Time-out Threshold τ - τ is used by a client to decide how long to wait for certification replies after sending out a certification request. Larger τ values can increase the possibility of success since the node waits longer for the CREPs to come back. But if there are not enough CREPs on their way back, the certification request will eventually fail and larger τ values can cause the node to wait needlessly. If τ is set too small, even when there are enough CREPs on their way back to the client, the client gives up too soon, discarding the CREPs on the way.

- Crypto Threshold *k* - *k* is the minimum number of CREPs required for a client to reconstruct the MOCA's full signature and render the certification request successful. If *k* is set low, a client only needs to collect a small number of *k* partial signatures to continue. Thus the success ratio increases and the packet overhead decreases, but at the same time an adversary only needs to compromise a small number of *k* MOCAs to compromise the framework. High *k* values can make attacks difficult, but the burden on clients and the cost in terms of packet overhead also increases since a client needs to contact a large number of MOCAs for any certification request.

- Unicast Threshold β - The unicast threshold β is the sum of the crypto threshold *k* and the margin value α. Larger α values make the framework more robust but limit optimizations because clients must have β instead of *k* cached routes to use the unicast-based approaches. Also a larger α value generates more overhead. When β-unicast is used, a larger α results in more unicast requests and replies. Also a larger α increases β, which reduces the probability of β-unicast being used and results in more flooding. Setting α to a low value makes it easier for a client to use unicast-based approaches, but may cause an excessive amount of certification failure due to the loss of too many CREQs or CREPs in the process.

By evaluating the results of the simulations, we provide some insight into how to configure the MOCA networks to achieve efficiency and availability at the same time.

5.1 Simulation Set-Up

We implemented our certification protocols in the ns-2 network simulator [2]. We test our protocol under two hypothetical scenarios. Consider a 1km by 1km battlefield with 150 or 300 friendly units including foot soldiers, jeeps, humvees, tanks and command vehicles. 30 MOCAs are deployed in both cases. 30 MOCAs represent 20% and 10% of the total nodes, which we believe to provide a reasonable number of MOCAs to support the ad hoc network. Each simulation is run for 10 minutes. One thing to note is that this scenario can be applied to other situations like a school field trip or a rescue operation. Although we use military examples to maintain consistency throughout the paper, none of our simulation factors depends on anything specific to military scenarios. Table 1 shows detailed simulation parameters.

We assume that any node that wishes to communicate with any other node in the network must first contact the MOCAs to either get the peer's certificate or to check the revocation status of the peer's certificate it acquired previously. The certification request pattern for the 150-node scenarios uses 100 non-MOCA nodes, each making 10 certification requests randomly distributed through the simulation timeline, for a total of 1000 certification requests. For the 300-node scenarios, 200 non-MOCA nodes make 10 certification requests each, adding up to a total of 2000 certification requests. Each requesting node makes one request per minute on average during the course of simulation. This is roughly 100 or 200 requests per minute and we believe that this is a reasonable number if not too pessimistic. Assuming each certification request precedes initiation of a new secure communication, starting one secure communication session per node per minute should be more than adequate for ordinary mobile nodes. Node movement follows the random waypoint mobility model implemented in the CMU Monarch extension [6] with pause times of 0 and 10

Total Number of Mobile Nodes	150 or 300
Number of MOCAs	30 or 50
Area of Network	1000m x 1000m
Total Simulation Time	600 seconds
Number of Certification Requests	10 requests each from 100 or 200 non-MOCAs
Node Pause Time	0, 10 seconds
Node Max. Speed	0, 1, 5, 10, 20 ms

Table 1: Simulation Parameters

seconds and maximum speeds of 0, 1, 5, 10 and 20 ms. Our simulation results show consistent results over different pause times, speed patterns and also number of MOCAs. Therefore in this section we only present the results for 0 second pause time, 10 m/s maximum speed and 30 MOCAs. Each line in Figures 1, 2, and 3 represents an average of three different runs with different mobility scenarios.

5.2 Flooding vs. Unicast

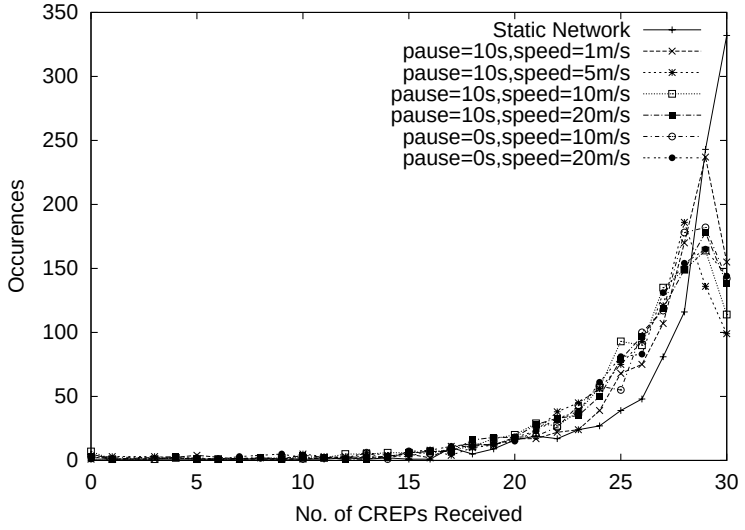

Figure 1: Flooding-based Certification Protocol

To evaluate the effects of employing unicast-based optimization, we first present results from a pure flooding-based approach. Figure 1 shows the number of CREPs received per CREQ under varying mobility. Under a stationary network, represented by the solid line, the flooding-based approach works very well. Almost all CREQs reach all 30 MOCAs and most CREPs make their way back to the client. The reason some of the CREPs get lost (there are many occurrences of nodes receiving 25 to 29 CREPs) is due to temporary network contention caused by the reverse packet storming effect generated by multiple CREPs traveling back to the client at almost the same time. As can be observed from the graph, a value of 15 or 20 for *k* can result in more than a 90% success ratio under all mobility scenarios and proves that flooding is indeed a very effective means of eliciting responses in ad hoc networks. More details on the flooding-base certification protocol can be found in [17].

Figure 2 shows the results from the Closest-Unicast approach with varying values of the unicast threshold β, with one line for the flooding-based approach for comparison. Consistent with the previous figure, the flooding line has a very high peak around 30, which is the number of MOCAs in the network. Each Closest-Unicast line has two peaks: one at β and another at 30. The peak around β shows that unicast is being used and works well.

Table 2 presents the total number of requests made as well as the number of requests using flooding and unicast. Note that the use of unicast CREQs decreases with higher β values, causing the height of unicast peaks in Figure 2 to

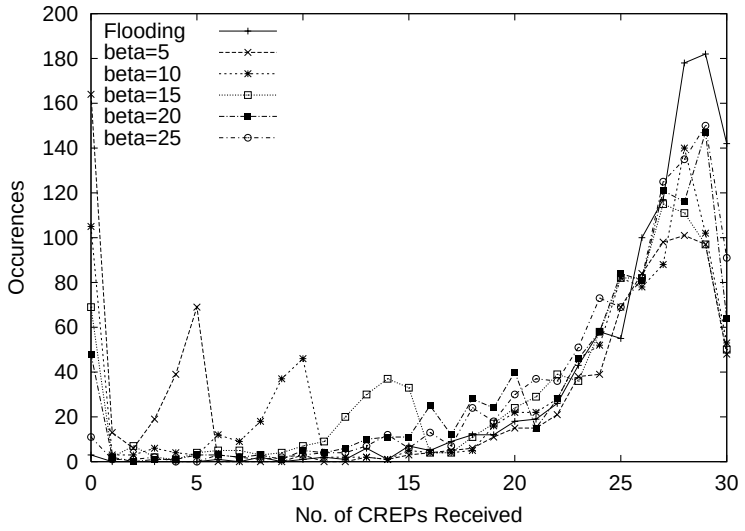

Figure 2: A Unicast-based Certification Protocol with varying β (using Close-unicast)

Table 2: Effect of β on Usage of Unicast

β	5	10	15	20	25	Flooding
Use of Unicast CREQs	337	241	200	172	128	0
Use of Flooding CREQs	663	759	800	828	872	1000
Total No. of CREQs	1000	1000	1000	1000	1000	1000

decrease as β increases. For higher β values, it is more likely that not enough routes to MOCAs will be cached, hence unicast-based optimization is used less often.

Figure 3 presents a comparison of the three unicast-based approaches. The unicast threshold β is set to 15, which can be translated into $k = 10$ with $\alpha = 5$ or $k = 12$ with $\alpha = 3$. We can observe that Closest-Unicast performs best with unicast CREQs. Closest-Unicast also induces the least overhead among the three unicast-based approaches as shown in the next subsection. For the rest of this section, we use Closest-Unicast as our example except when providing a comparison between different unicast approaches.

5.3 Packet Overhead

We evaluate communication overhead, as measured by the total number of control packets used for certification services. Table 3 shows the overhead from flooding and various unicast-based approaches under varying unicast thresholds, β. Generally, unicast-based approaches save 5 to 20 percent of control packet overhead. As the node chooses unicast more aggressively with lower β, the savings are increased. Note that when β is 20 or 25, there is little improvement over flooding. In these cases, β is very high and unicast is not used often since many nodes do not have enough cached routes to MOCAs. This causes most certification requests to fall back to flooding, generating a similar amount of overhead as in flooding. Also, the amount of traffic generated by β unicast CREQs increases as β increases, adding more overhead. In a more reasonable scenario of β =15 or less, unicast-based approaches save between 15 to 30 percent as compared to flooding. Setting β as low as possible results in the best improvements in overhead but has the adverse effect of implicitly lowering the upper bound of crypto threshold *k* to a very small number, endangering the security of the whole framework as described in Section 2.

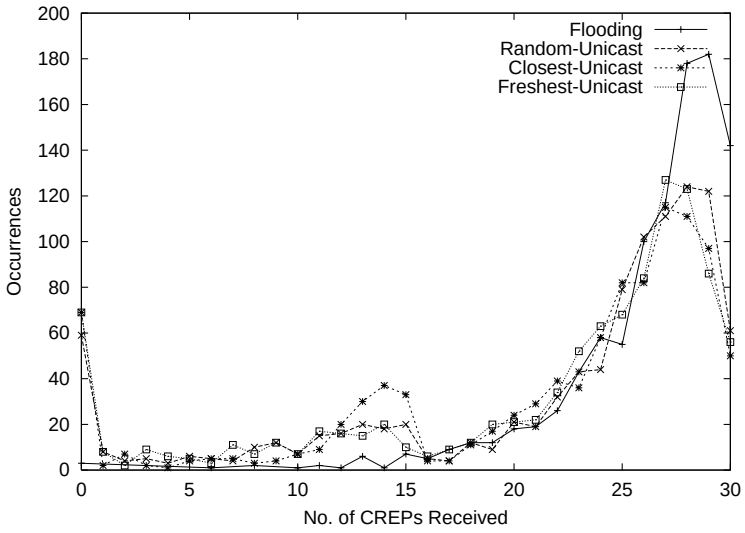

Figure 3: Comparison among Unicast-based Optimizations, β = 15

5.4 Certification Delay

The most frequent use of a certification service is to acquire the communicating peer's public key certificate. The delay to get the certification service is added to the start-up latency of any secure communication relying on PKI.

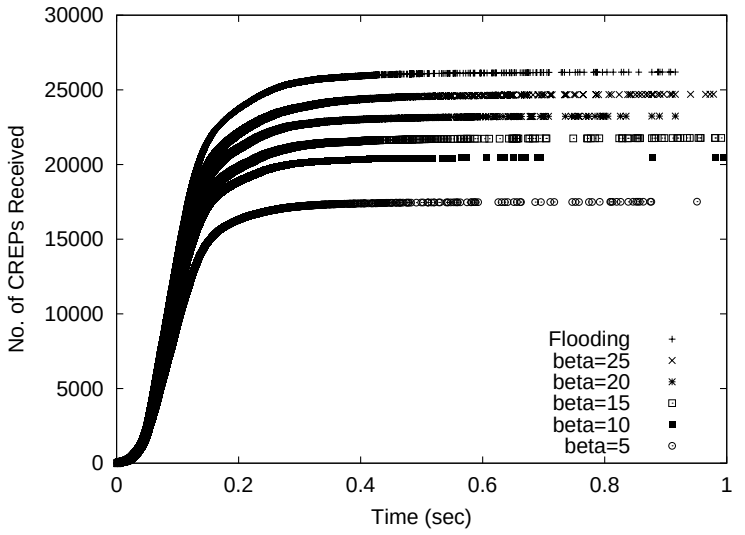

Figure 4: No. of CREPs received over the course of time, using Closest-Unicast

Figure 4 shows the distribution of arrival times of CREP packets with the Closest-Unicast approach with varying β under a moderate mobility pattern of 0 pause time and 10 ms maximum speed. Also, a line for flooding is presented for comparison. Over all cases, the lines flatten out quite quickly, indicating that a client can expect to receive most pending CREPs within 0.3 seconds from the time of certification request. If the client does not collect enough CREPs within that time, the chances are very slim that enough CREPs are in in-flight to arrive later and fulfill the certification request. Based on a appropriately chosen time-out threshold τ, a client can operate efficiently without wasting time. To clarify the choice of 0.3 seconds, Figure 5 shows a normalized view of Figure 4. The choice between flooding and unicast-based optimizations or the choice between different β values does not affect the timing behavior. This indicates that only the density of MOCA nodes affects timing behavior. If MOCAs are densely deployed, a client has

Table 3: Packet Overhead, n = 30

Number of Packets	CREQ	CREP	Total	Ratio to Flooding (%)
Flooding	119642	77959	197601	100
Random-Unicast				
β = 5	84230	54337	138567	70.1
β = 10	97132	61920	159052	80.5
β = 15	105599	67276	172875	87.5
β = 20	110217	69903	180120	91.2
β = 25	114805	73321	188126	95.2
Closest-Unicast				
β = 5	83174	54492	137666	69.7
β = 10	96781	62258	159039	80.5
β = 15	103749	66626	170375	86.2
β = 20	108543	68821	177364	89.8
β = 25	113859	73204	187063	94.7
Freshest-Unicast				
β = 5	85668	54966	140634	71.2
β = 10	97578	62470	160049	81.0
β = 15	105818	67285	173103	87.6
β = 20	111637	70619	182256	95.2
β = 25	114807	73454	188261	95.3

a better chance to discover enough MOCAs faster.

To get a better understanding of this graph, Figures 6 and 7 show a more detailed look at two of the lines from Figure 4.

Figure 6 shows the success ratios for different τ and *k* for the flooding line in Figure 4. When $\tau = 0.1$ seconds, the success ratio drops rapidly as *k* increases. As τ increases, the success ratios with higher *k* values approach a stable value. These results support the choice of 0.3 sec for τ. Similar trends can be observed from Figure 7 for Closest-Unicast. The success ratios of higher *k* values stabilize faster than in Figure 6. For example, the success ratios do not change very much after 0.2 seconds, because the MOCAs are chosen based on the hop count in Closest-Unicast and the CREPs will arrive earlier from the close MOCAs.

One thing noticeable from the two detailed looks at the success ratio is that α plays an important role in determining the success ratio within a given τ. For example, the set of leftmost data points in Figure 6 represents the success ratio with τ set to 0.1 seconds. Each point in the set represents the success ratio under varying values of *k*. For example, when *k* = 1, which is practically a replicated CA case, the success ratio within 0.1 seconds is almost 98%. In comparison, when *k* is set to 10, the success ratio within the same time-out threshold drops down to little less than 70%. The same general trend can be observed over all sets in Figure 6 and also with the unicast-based approach in Figure 7.

These detailed graphs can be helpful when deciding an adequate τ for a given *k*. For example, if *k* is set to 15 out of 30 MOCAs for a network using Closest-Unicast, τ can be chosen to be 0.2 seconds to maintain higher than 90% success ratio.

6 Future Work

Our future work is in two directions. First, we are planning to optimize the current certification protocol to be more efficient and adaptable. Second, we are investigating possible extensions of our framework to address the network partition problem and to integrate with other security services for ad hoc networks.

Our current β-unicast approach only exploits information in the local routing cache of a client. One potential extension is to let a node browse into neighboring nodes' routing tables. For example, a node may have one or two

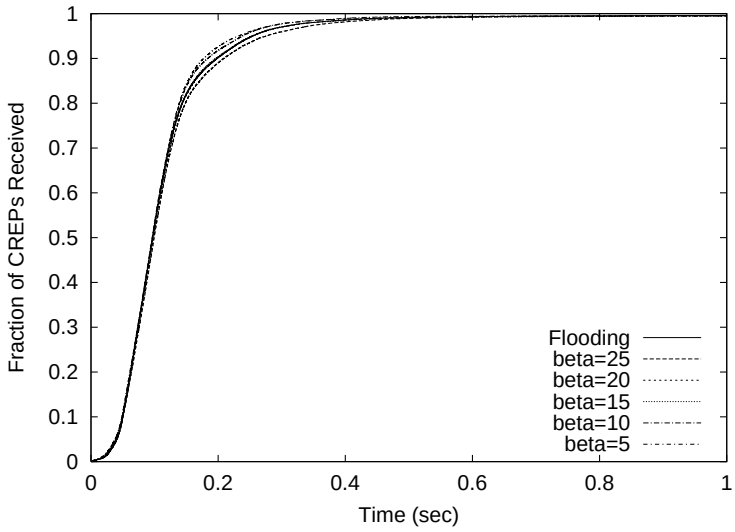

Figure 5: No. of CREPs received over the course of time, Normalized, using Closest-Unicast

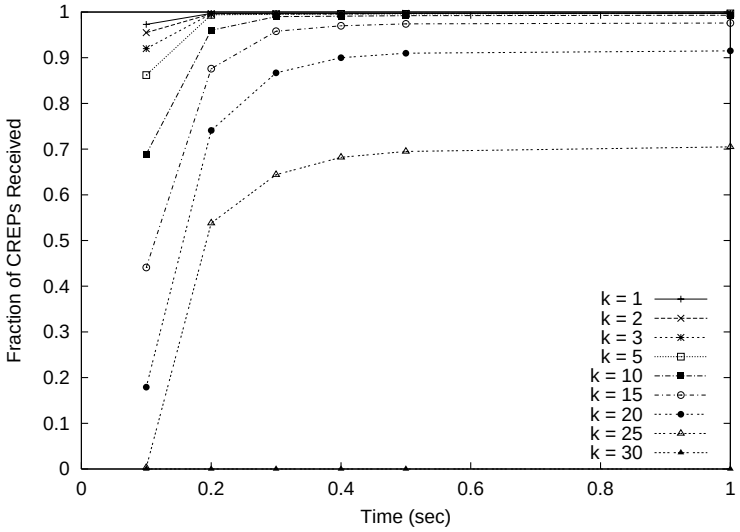

Figure 6: Success Ratio with Flooding

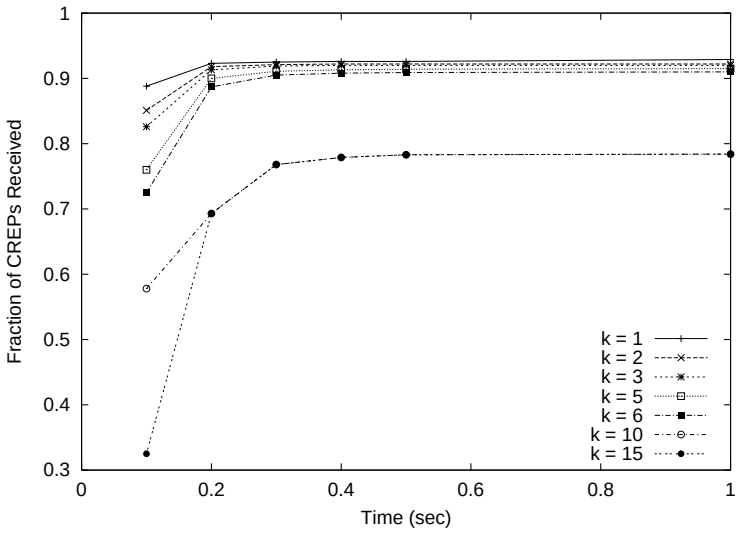

Figure 7: Success Ratio with Closest-Unicast

cached routes short of β and will have to fall back to flooding. If the node can peek at a neighbor's routing tables and find new cached routes, it can enable β-unicast and avoid flooding. The potential overhead from this approach would be the extra communication required between neighbors to exchange the information in routing tables. Whether the benefit would surpass the overhead is an interesting question to investigate.

All unicast-based approaches in our current protocol do not take into account the direction of CREQs. For an extreme case, all the MOCAs picked by our unicast approach could reside on one side of the network from the requesting node. Then it is possible that all the CREQs are sent into one direction sharing the same next hop nodes, potentially causing unnecessary contention that leads to a failure or at least delayed responses. One possible solution for such a situation is to utilize the next hop field in the cached routing table entries. For example, by selecting a set of MOCAs with all different next hops, we can expect to have a spatial load balancing effect in that each CREQ will go out in different directions.

Another interesting direction we plan to investigate is dynamic adjustment of the time-out threshold τ. As presented in Section 5, τ can be selected based on the MOCA density in the neighborhood, which is likely to change as the nodes move around in the network. We plan to investigate the mechanisms to adjust τ to reflect the updated perception of the new neighborhood, hence reducing the certification delay to a minimum.

While we have designed a PKI framework that provides balanced support for security and availability, we cannot avoid the inherent problem of ad hoc networks: unstable connectivity. In a pathological case, if the network is partitioned and there are less than *k* MOCAs available in a partition, it is simply not possible to get a certification service. Although we do not expect to see this kind of problematic situation too often or for too long a time period, if this happens our approach became powerless. To provide certification support for such scenarios, we are currently developing an extension of the MOCA by introducing a hybrid approach of MOCA and the PGP "Web of Trust" model. In the extended MOCA (EMOCA) framework, any node certified by *k* MOCAs will have the capability to act as a delegate of MOCAs to authenticate and issue certificates to new nodes or yet uncertified nodes in the network. If a node wishes to get a certificate but cannot reach enough MOCAs, it can then contact any nearby certified nodes and request a *temporary certificate*. Any already certified node can issue a temporary certificate based on its own authentication of the new node. This temporary certificate carries relatively small confidence compared to the one issued by MOCAs but still can be used as a temporary means for confirmed identity. Conceptually, a certificate issued by MOCAs can be considered as the voucher for confirmed identity by trusted entities (i.e. MOCAs). A temporary certificate serves a similar goal but with a smaller confidence value since the vouching entity is not a trusted entity but only a confirmed member of a network. In our preliminary investigation, we have discovered several interesting features of this hybrid approach and are currently studying the interaction between MOCAs and the delegates and their effects on performance and security.

There are many interesting and promising security services and applications that can be deployed in ad hoc net-

works using the support of PKI. For example, some secure ad hoc routing protocols that assume the existence of PKI support can readily utilize our framework [16, 14]. However, it is yet unclear how these different security services and applications will fit with each other. We plan to study how our approach can be integrated with other security services or applications and what kind of effects will occur.

7 Conclusion

In this paper, we present a practical key management framework for ad hoc wireless networks. We clarify the necessity and the problem of providing a PKI framework for ad hoc networks and identify the requirements for such a framework. Based on our observation of the potential heterogeneity among mobile nodes, we provide an intelligent way to pick a set of CA nodes. These selected secure nodes are called MOCAs and share the responsibility of collectively providing the CA functionality for an ad hoc network using threshold cryptography. To minimize the usage of scarce resources in mobile nodes, we develop a set of efficient and effective communication protocols for mobile nodes to correspond with MOCAs and receive certification services. Our simulation results show the effectiveness of our approach and we provide some insights into the configuration of such security services in ad hoc networks.

References

[1] Thawte, Inc. Company homepage available at http://www.thawte.com/.

[2] The Network Simulator - NS-2. Available at http://www.isi.edu/nsnam/ns/.

[3] VeriSign, Inc. Company homepage available at http://www.verisign.com/.

[4] A. Shamir. How to Share a Secret. *Communications of the ACM*, 1979.

[5] A. Arnes. Public key certificate revocation schemes. Available at http://citeseer.nj.nec.com/arnes00public.html.

[6] J. Broch, D. A. Maltz, D. B. Johnson, Y. Hu, and J. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In *Proceedings of IEEE/ACM MOBICOM 98*.

[7] C. E. Perkins and E. M. Royer. Ad-hoc On-Demand Distance Vector Routing. In *The Second IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, USA, February 1999.

[8] Y. Frankel and Y. G. Desmedt. Parallel Reliable Threshold Multisignature. Technical Report TR-92-04-02, Univ. of Wisconsin–Milwaukee, 1992.

[9] Janne Gustafsson, Janne Lassila, and et al. Pki-security in mobile business - case: Sonera smarttrust. Available at citeseer.nj.nec.com/466933.html.

[10] J.-P. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. In *Proceeding of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC 01)*, 2001.

[11] J. Broch and D. B. Johnson. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks. IETF Internet Draft, October 1999.

[12] J. Macker and M. Corson. Mobile ad hoc networking and the IETF. *Mobile Computing and Communications Review*, 1998.

[13] J. Macker and S. Corson. Mobile Ad-hoc Networks (MANET) Charter. IETF Working Group.

[14] K. Sanzgiri and B. Dahill and B. Levine and C. Shields and E. Belding-Royer. A Secure Routing Protocol for Ad Hoc Networks. In *Proceedings of 2002 IEEE International Conference on Network Protocols (ICNP)*, November 2002.

[15] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks. In *Proceedings of ICNP '01*.

[16] M. Zapata. Secure Ad Hoc On-Demand Distance Vector (SAODV) Routing. IETF MANET Mailing List, Message-ID:3BC17B40.BBF52E09@nokia.com, Available at ftp://manet.itd.nrl.navy.mil/pub/manet/2001-10.mail, October 8 2001.

[17] S. Yi and R. Kravets. Practical PKI for Ad Hoc Wireless Networks. Technical Report UIUCDCS-R-2002-2273/UILU-ENG-2002-1717, University of Illinois at Urbana-Champaign, May 2002.

[18] V. Shoup. Practical Threshold Signatures. In *Theory and Application of Cryptographic Techniques*, pages 207–220, 2000.

[19] Sthephen Kent and Tim Polk. IETF Public-Key Infrastructure Working Group Charter. Available at http://www.ietf.org/html.charters/pkix-charter.html.

[20] W. Diffie and M.E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 1976.

[21] L. Zhou and Z. J. Haas. Securing Ad Hoc Networks. *IEEE Network Magazine*, November 1999.

[22] L. Zhou, F. Schneider, and R. van Renesse. Coca: A secure distributed on-line certification authority. Technical Report, Cornell University.