NetTRUST: mixed NETworks Trust infrastRUcture baSed on Threshold cryptography

Mawloud Omar, Yacine Challal, Abdelmadjid Bouabdallah

NetTRUST: mixed NETworks Trust infrastRUcture baSed on Threshold cryptography

Mawloud Omar
ReSyD, Bejaia University, Algeria.
mawloud.omar@gmail.com

Yacine Challal
Heudiasyc Lab., UTC, France.
ychallal@hds.utc.fr

Abdelmadjid Bouabdallah
Heudiasyc Lab., UTC, France.
bouabdal@hds.utc.fr

***Abstract*—The proliferation of network technologies (wired, cellular, ad-hoc, etc.) leads to many different network architectures. These different architectures cohabitate to provide services and contents to end customers. In order to secure services in such mixed networks, it is necessary to rely on a homogeneous trust model. The trust model must define trust relationships between the mixed architecture actors, provide elementary ingredients to secure top level services, and guarantee the security service availability. In this paper, we propose a trust infrastructure for mixed networks architectures. The model uses two particular certification authorities, which ensure X$509v3$ certificates management: the central certification authorities (CCA) are tied to the portions of the network having a pre-existent communication infrastructure (such as wired networks, cellular networks, etc.), and mobile certification authorities (MCA) which are on the ad-hoc portion of the network. The MCA servers emulate the certification authority role using a (k, n) threshold cryptography scheme, and the CCA servers delegate the role of certification to the MCA servers by using a (t, m) scheme of threshold cryptography. This solution is decentralized and partially distributed, supports the nodes mobility and the failure of, up to $n-k$, among n MCA servers. The simulation results and the performance evaluation prove the adequacy of this solution to mixed networks architectures.**

***Keywords*—Trust Models, Public-Key Certificate Management, PKI, Threshold Cryptography, Mixed Architecture.**

I. Introduction

THE massive use of applications over the Internet for commercial ends drove many research efforts towards offering security services such as authentication, integrity and data confidentiality, in order to enlarge the scope of users having confidence in online business oriented applications. On the other hand, the requirement for more mobility made very widespread the concept of infrastructure-less networks such as ad-hoc networks. An ad-hoc network comprises a group of wireless nodes which temporarily form a network without pre-existing infrastructure. Purely ad-hoc networks have few applications (generally restricted to military applications), but the combination of ad-hoc networks with infrastructure based networks, generally called mixed networks (or hybrid networks), provides a large spectrum of applications and facilities [7], [14] (cf. Figure-1): companies could install a wireless access network for their itinerant personnel, the operators could increase their coverage ratio while reducing deployment costs, and so on.

The trust model provides a framework for the construction and the administration of trust relation between the nodes in a network [17]. According to ITU−T, the trust term is defined as follows: "Entity A trusts entity B when A assumes that B will behave exactly as A expects" [18]. The trust relation in the majority of security architectures is based on TTP (Third Trust Party), where TTP is a special entity trusted by all the other entities. There are also certain schemes that rely on the collaborative effort between nodes to establish trust in the network. There exist, also, schemes based on the reputation concept, where trust changes according to the nodes behavior.

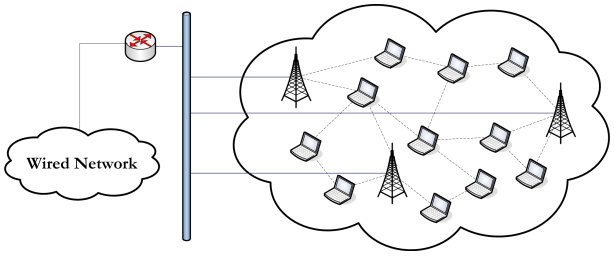

Fig. 1. Mixed Networks Architectures.

The development of any security service requires preliminary knowledge of the subjacent trust model. In the literature, there are several trust models that can be classified into two main approaches: the models based on TTP, like PKI (Public Key Infrastructure) [23], or Kerberos [30]. There are other models without or slightly dependent on TTP. These models are more appropriate to the ad-hoc networks, like the distributed models [28], the models based on trust propagation through a trust graph (PGP [27]), or the failure tolerant models, like those based on threshold cryptography [12], [21].

In this paper, we propose a trust infrastructure for mixed networks architectures. The model uses two particular certification authorities, which ensure X$509v3$ certificates management: the central certification authorities (CCA) are tied to the portions of the network having a pre-existent communication infrastructure (such as wired networks, cellular networks, etc.), and mobile certification authorities (MCA) which are on the ad-hoc portion of the network. The MCA servers emulate the certification authority role using a (k, n) threshold cryptography scheme, and the CCA servers delegate the role

of certification to the MCA servers by using a (t, m) scheme of threshold cryptography. This solution is decentralized and partially distributed, supports the nodes mobility and the failure of, up to $n-k$, among n MCA servers. The simulation results and the performance evaluation prove the adequacy of this solution to the mixed networks architectures.

The remainder of this paper is organized as follows. In section-II, we give an overview of related works. In section-III, we present our trust infrastructure for mixed networks architectures. Then we present in section-IV the simulation and performance evaluation results. We end this paper with a general conclusion.

II. Related Work

This section surveys trust models in both wired networks and ad-hoc networks. We classify trust models in three principle classes: centralized models, partially distributed models, and completely distributed models (cf. Figure-2).

A. Centralized Models

Since its birth more than two decades ago [33], public-key cryptography has been recognized as one of the most effective mechanisms for providing security services, such as authentication, digital signatures and encryption. The digital certificates management is a key factor for the successful wide-spread deployment of public-key cryptography. PKI (Public Key Infrastructure), a centralized infrastructure for digital certificates management, was introduced exactly for this purpose [11], [23]. PKI is based on TTP (Third Trust Party) called CA (Certification Authority), the trusted entity in the system. The role of CA is to certify the association between an entity and the corresponding public-key. The success of PKI depends on the security and availability of the CA to the principals in a system 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, etc. PKI has been deployed for wired networks and some infrastructure-based wireless networks [8]. Since good connectivity can be assumed in these networks, the main thrust of research in such environments has focused on the security and the scalability of the certification authority to handle a large number of requests.

Kerberos [29], [30] is a centralized trust model, it relies on a TTP, referred to a KDC (Key Distribution Center). $Alice$, a Kerberos principal, and Bob, a Kerberized service, establish shared secrets with the KDC. Kerberos Service acts as the trust reference in the system. Kerberos shares different key-secrets with each entity in the network, and it creates session keys which are used among users and servers in order to establish the communications in security.

B. Partially Distributed Models

In [25], L. Zhou et al. proposed, for ad-hoc networks, a partially distributed certification authority, relying on threshold cryptography techniques. The service is distributed to a particular group of nodes: *servers*, *combiners*, and a *dealer*. The servers (and combiners) sign digital certificates for users. The dealer is a particular server which knows the completely private-key certification authority.

In [12], L. Zhou et al. proposed COCA (Cornell On-line Authority Certification), for local area networks and Internet, a certification authority which combines at the same time the failures tolerance and security. With $3t + 1$ COCA servers, only t can be defective or compromised. The system COCA integrated a particular proactive protocol to update certification service private-key shares to tolerate the *mobile adversaries*.

In [8], S. Yi et al. proposed, for ad-hoc networks, a solution based on public-key infrastructure. The model rests on a scheme of (k, n) of threshold cryptography to distribute the certification authority role to a group of servers, called MOCA servers (MObile Certification Authority). The communication established between users and servers is one-to-many-to-one (*manycast*). MOCA integrate a certification protocol MP (Moca Protocol) for certificates management.

In [4], A. Pirzada et al. proposed KAMAN (Kerberos assisted Authentication in Mobile Ad-hoc Networks). KAMAN is based on the time-tested and deployed Kerberos protocol, and provides secure extensions to support the more challenging demands of ad-hoc networks. KAMAN migrates a number of features from the traditional, wired Kerberos environments to the ad-hoc environment, including the prevention of node identity forgery, the detection of replay attacks, establishment of secure channels, mutual endpoint authentication, and the secure distribution of provisional session keys amongst replicated servers. KAMAN has been designed for hostile environments, in which the presence of malicious nodes and the likelihood of physical node capture is relatively high.

In [2], J. Luo et al. proposed DICTATE (DIstributed CerTification Authority with probabilisTic frEshness for ad-hoc networks), an public-key architecture for ad-hoc networks, based on the PILOT system layers services (ProbabilistIc Lightweight grOup communication sysTem) [5]. PILOT is composed of two layers containing a multicast protocols and services collection: RDG (Road Driven Gossip) [26], R^2DG (Applicable RDG), PAN (Probabilistic quorum system for Ad-hoc Networks) [19]. In DICTATE, they suppose the presence of a certification authority mother, called mCA (mother Certification Authority). Nodes can often collectively be isolated from the mCA, but always have the need for certification authority access. Thus, the mCA delegates the service to a group of servers, called dCA (distributed Certification Authority), during the insolation period.

In [3], B. Wu et al. proposed SEKM (Secure and Efficient Key Management in mobile ad-hoc networks). In SEKM, the trust of the central authority is distributed to a subset of nodes, which could be nodes with normal or better equipment. SEKM is designed to provide efficient share updating among servers and to quickly respond to certificate updating. For efficiency, only a subset of the server nodes initiates the share update phase in each round. A ticket based scheme is introduced for efficient certificate updating.

In [1], S. Raghani et al. proposed for the ad-hoc networks, solution based on PKI. The model rests on threshold

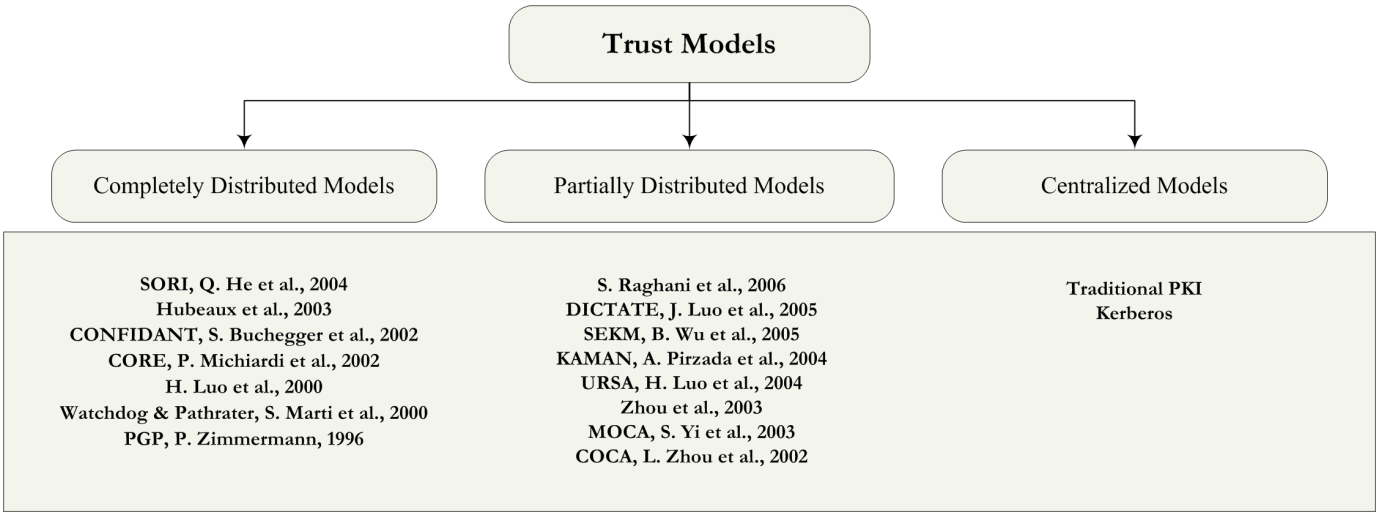

Fig. 2. Taxonomy of Trust Models.

cryptography to distribute the certification authority role. The proposed model provides a dynamic support for distributed certification authority by allowing it to dynamically adjust the threshold value when required and thereby resulting in reduction in certification service delays.

C. Completely Distributed Models

PGP (Pretty Good Privacy) [27] is a completely distributed model which was created, by P. Zimmermann, initially for Internet. PGP is an alternative to the PKI based on trusted authorities, provides practical security to protect low value communications, such as emails. PGP is based on referral certification, which allows multiple users to ”recommend” a certain user by signing certificates of its public-key. PGP adopts a system, called ”Web of Trust”. It consists of an establishment of a distributed key management. The principle interest, in this model, lies in the absence of a central authority. However, this scheme is not perfectly secure because, for example, dishonest users may issue false certificates to cheat other users.

In [10], J. Hubaux et al. proposed, for ad-hoc networks, a fully self-organized public-key management system that allows users to generate their public/private key pairs, to issue certificates, and to perform authentication regardless of the network partitions and without any centralized services. Furthermore, the system does not require any trusted authority, even in the system initialization phase.

In [21], H. Luo et al. proposed, for the ad-hoc networks, a completely distributed certification authority model, with the use of threshold cryptography. The model distributes shares to all entities at the time when they join the network. Trust is established by the assumption that all the nodes must supervise the direct neighbors behavior, and maintain their own CRL (Certificate Revocation List). If a node discovers that one of its neighbors is incorrect, it adds its certificate to the list of revocations and diffuses through the network an *accusation*. If the certificate of accusatory is revoked, the accusation is ignored. Otherwise, the node is marked suspect by all the nodes receiving the accusation.

We find, also, trust models based on reputation mechanisms such as: Watchdog and Pathrater [22], CONFIDANT [16], CORE (COllaborative REputation mechanism) [17], and SORI (Secure and Objective Reputation based Incentive design) [6].

III. Our Architecture: NetTRUST−mixed NETworks Trust infrastRUcture baSed on Threshold cryptography

In this section, we present our architecture NetTRUST, for mixed networks.

A. Motivation and Objectives

We consider a mixed network as an ad-hoc network, which is connected to an infrastructure based network (i.e. a network equipped with an infrastructure: wired or cellular) through access points. In the ad-hoc network part, the nodes can be mobile. Thus, the network can undergo disconnections and partitioning because of nodes mobility.

The basic objective is to define a trust infrastructure which satisfies the following properties:

- The ad-hoc network part is characterized by the dynamicity of nodes. So, the architecture must be adapted to support the mobility of users.
- The ad-hoc network part is vulnerable and subject to nodes disconnections and network partitioning. So, it is required to ensure the availability of the security services.
- The security services must be guaranteed without a central entity.
- The architecture must guarantee an optimal delay of requests treatment.
- The architecture must be adapted to the scale factor, i.e. have the capacity to support the evolution of users' number in the system.

B. Overview of NetTRUST

The centralized trust architectures suffer from the single point of failure property. Therefore, in our infrastructure we reduce this vulnerability through a distributed solution, where several servers ensure the certification authority services. Our architecture NetTRUST distributes the role of certification authority among two sorts of servers:

- Mobile authorities MCA (Mobile Certification Authority).
- Central authorities CCA (Central Certification Authority).

The MCA servers share the certificates signature role by using threshold cryptography. Based on this technique, the private-key of the certification authority is divided into n shares. Each MCA server holds one share. In order to sign certificates for users, each MCA server generates a partial signature by using its private-share. A coalition of k MCA servers are able to generate a complete certificate signed with the certification authority private-key. Therefore, users should request all MCA servers and combine their partial certificates. NetTRUST uses a (k, n) threshold cryptography scheme. It means that in order to obtain a certificate we must have at least k partial certificates signed by k MCA servers.[1] An example of a threshold signature technique is the RSA signature scheme defined by M. Hwang et al. in [9].

In NetTRUST, the CCA servers perform two principal functions. The first is to behave exactly as a usual certification authority. Therefore, their responsibilities will be: sign, revoke, and update the certificates for users. The second task is the control of MCA servers. In the initialization phase, the CCA servers jointly create and distribute n private-shares of the certification authority private-key to the MCA servers. Hence, the MCA servers will emulate the certification authority role for users that have not access to the wired part of the network. In the ad-hoc network, the certification protocol is executed among the n MCA servers, which can be safe or compromised. A compromised MCA server can stop or suspend its execution and/or exclude access to the data saved in the server. The compromising of $n-k$ MCA servers, can be done in a limited time called *window of vulnerability* [12]. Thus, in NetTRUST, the CCA servers periodically update the MCA servers' private-shares. The updating period must be estimated to protect at least $n-k$ MCA servers. The global architecture of NetTRUST (cf. Figure-3) is composed of two subsystems:

- DTCS (Delegation Threshold Cryptography System).
- CTCS (Certification Threshold Cryptography System).

In NetTRUST, the MCA and the CCA servers execute the same certificates construction policy. The partial certificates issued by each MCA server are similar. The role of the user consists, only, of checking the correctness of certification authority signature. If users, in the ad-hoc network, have no access to the CCA servers then, they must request and combine at least k partial certificates from the MCA servers.

[1]More then $n - k$ MCA servers must be compromised to be able to compromise all the system.

Here are the various notations used in this paper:

CCERT	Complete CERTificate
PCERT	Partial CERTificate
DCERT	Delegation CERTificate
K_i, K_i^{-1}	i's public-key, i's private-key
$\{\text{msg}\}^{K_i}$	message msg encrypted with i's public-key
$\{\text{msg}\}^{K_i^{-1}}$	message msg signed with i's private-key

C. Certificates Format

In NetTRUST, we use the X509$v3$ certificate format [24]. The certificates contain: version, serial number, signature algorithm name, sender name, holder name, public-key, validity period, attributes, extensions, and the certification authority signature.

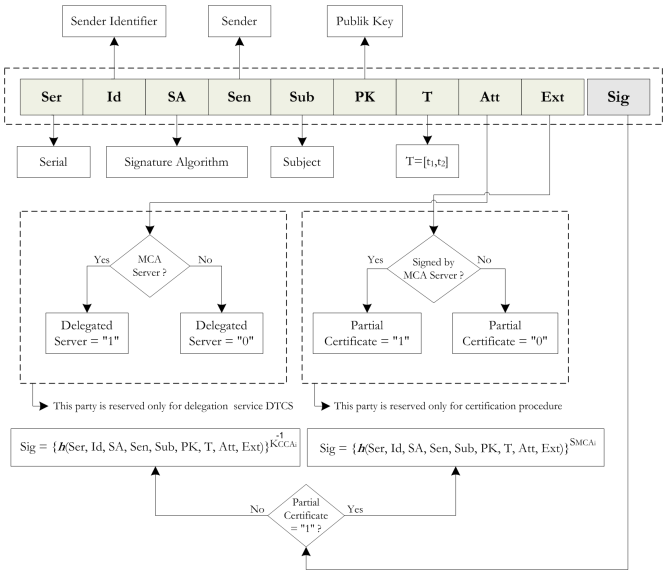

Fig. 4. Certificates Format in NetTRUST.

The certificates signature is done by one CCA server or a coalition of k MCA servers. Thus, the signer must indicate the nature of the certificate in order to eliminate conflicts on the signature checking operation. Hence, the users can make the difference between the *certificate* and the *partial certificate*. NetTRUST employs the field *extensions* to save signer indications. Initially, the user must observe the *extensions* field. If the certificate was signed by a MCA server (*partial certificate*), the user must save it locally. Upon receiving k partial certificates, the user will be able to combine them and recover the complete certificate signed by the service private-key. In order to perform the delegation operation between CCA and MCA servers, NetTRUST uses the *attributes* field. Each MCA server must have a delegation certificate (DCERT), signed by the DTCS (cf. Section-III-D). The user can trust a MCA server, only if the server has the rights of the certification function. Thus, the user must observe the *attributes* field on the delegation certificate of the MCA server, in order to trust the partial certificate delivered from it.

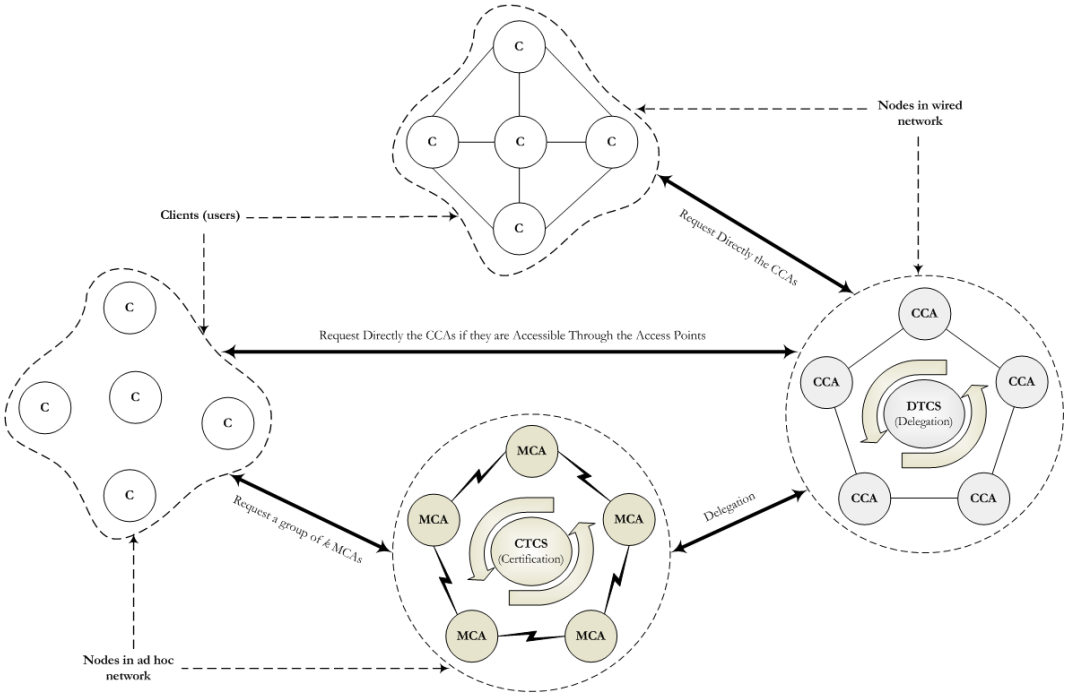

Fig. 3. Architecture of NetTRUST.

D. DTCS (Delegation Threshold Cryptography System)

During network initialization, CCA servers cooperate to share the delegation service by creating jointly the delegation private-key. Then, the MCA servers obtain shares value of the certification authority private-key from the CCA servers. We use threshold cryptography in DTCS in order to distribute the delegation service among CCA servers. Thus, a scheme of (t, m) is used to allow any coalition of t CCA servers in the infrastructure-based part of the network to jointly perform delegations to the MCA servers. A coalition of t CCA servers forms a DTCS. In NetTRUST, we can insert new MCA servers, which they must get a delegation certificate signed by the DTCS (cf. Figure-5).

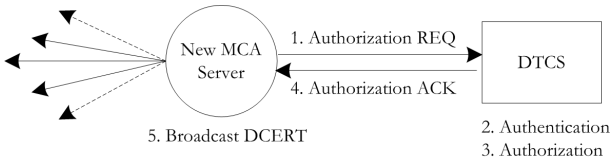

Fig. 5. Delegation Service.

The protocol begins with a new MCA server (noted NMCA) wishing to join the CTCS (cf. Section-III-E). Therefore, NMCA must send an *Authorization Request* ($AReq$) to the DTCS. The DTCS authenticates the NMCA and checks if it has rights to join the certification service. The DTCS returns an *Authorization Acknowledgment* (AA) specifying a successful or unsuccessful authorization. If successful, the authorization acknowledgment will contain a delegation certificate DCERT_{NMCA} and a private-share S_{NMCA}. Then, the NMCA server broadcasts a *Delegation Declaration* (DD) containing DCERT_{NMCA} for all nodes in the system. Here is the overview of the protocol:

- $NMCA \longrightarrow DTCS : AReq$
- $DTCS \longrightarrow NMCA :$
  $AA = \{S_{NMCA}, DCERT_{NMCA}\}^{K^{-1}_{DTCS}}$
- $NMCA{-}broadcast : DD = \{DCERT_{NMCA}\}^{K^{-1}_{NMCA}}$

E. CTCS (Certification Threshold Cryptography System)

We use the threshold cryptography in CTCS, in order to distribute certificates to end users. A (k, n) scheme allows any k of n MCA servers to jointly perform a certificate signature. Therefore, any coalition of less than k MCA servers is unable to perform the signature operation.

In order to get a certificate, the users must broadcast a request in the network (cf. Figure-6). Each MCA server generates a partial certificate using its private-share and sends it to the user. Thus, when the user obtains a subset of k correct partial signatures, he becomes able to generate a complete certificate (CCERT) signed by the CTCS private-key.

Figure-7 illustrates an example of a $(2, 3)$ threshold cryptography scheme in CTCS. In this case, the system contains three MCA servers and the certification threshold is equal to 2. Each MCA_i server holds a certification service private-share. Thus, each MCA_i can generate a partial certificate PCERT_{MCA_i} using the S_{MCA_i}'s private-share. A user broadcasts a certification request in the network. Then, MCA_1 and MCA_3 generate

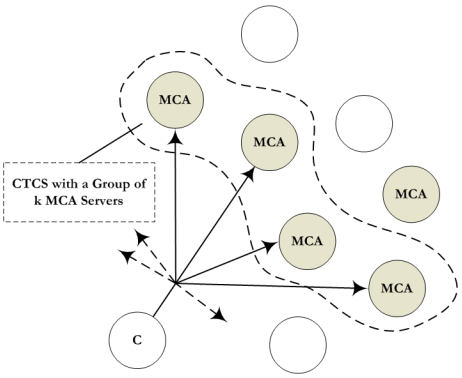

Fig. 6. Certification Request Broadcasting.

partial certificates: PCERT_{MCA_1} PCERT_{MCA_3}, and they send them to the user. We assume that MCA_2 failed to generate PCERT_{MCA_2}. Nevertheless, the user is able to generate the complete certificate CCERT, because only two correct partial signatures are required to have the complete certificate.

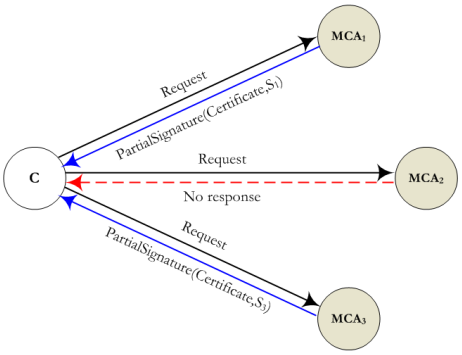

Fig. 7. Example of $(2, 3)$ CTCS Scheme.

Moreover, NetTRUST resists against compromised MCA servers. In the case a compromised MCA server generates an invalid signature, the signature checking fails, and the user must reselect another subset of k partial certificates.[2]

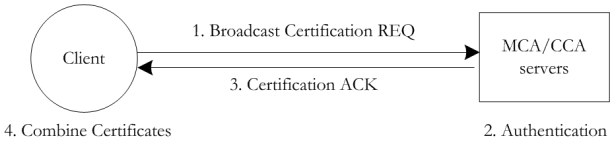

Fig. 8. Certification Service.

During network initialization, CCA servers generate RSA keys [32] for the CTCS: $\{(e, \eta),(d, \eta)\}$, where e is RSA public-exponent, d is RSA private-exponent and η is the RSA modulus. Then, the CCA servers define a $k-1$ degree polynomial function $f(x) = \alpha_0+\alpha_1 x^1+...+\alpha_{k-1}x^{k-1}$, where α_0=d and $\alpha_1, \alpha_2, ..., \alpha_{k-1}$ are random values. k represents the threshold value of the CTCS. The CCA servers initialize n MCA servers by providing them the private-shares of the certification authority service private-key. Each MCA_i receives its private-share: S_{MCA_i}=$f(\text{ID}_{MCA_i}) \bmod \eta$, where ID_{MCA_i} is a unique identifier of MCA_i in the network.

The certification protocol begins with a user (noted Client) wishing to request a certificate. The user must broadcast a *Certification Request* ($CReq$) in the network. Here is the overview of the certification protocol:

- $Client - broadcast : CReq$
- $CCA_i \longrightarrow Client : CRes = \{CCERT\}^{K^{-1}_{CCA_i}}$
- $MCA_i \longrightarrow Client :$
  $CRes = \{PCERT^{S_{MCA_i} l_{MCA_i}}\}^{K^{-1}_{MCA_i}}$

Each MCA server (respectively, CCA server) returns a *Certification Response* ($CRes$) which contains the partial certificate (respectively, the complete certificate) signed by the MCA's private-share (respectively, CCA's private-key). If no CCERT is received[3], the user must combine the partial certificates received from MCA servers. For threshold signatures combination we apply the *k−bounded coalition offsetting algorithm* proposed by J. Kong et al. in [20] (in order to construct the complete certificates). A coalition of k MCA servers $\{\text{MCA}_1, \text{MCA}_2, ..., \text{MCA}_k\}$ can recover the secret via Lagrange interpolation:

$$d = \sum_{i=1}^{k} S_{MCA_i} l_{MCA_i} mod \eta$$

Where l_{MCA_i} are the Lagrange coefficients defined as:

$$l_{MCA_i} = \prod_{j=1,\, j\neq i}^{k} \frac{ID_{MCA_j}}{ID_{MCA_j} - ID_{MCA_i}}$$

F. Private-Shares Update

Periodically, in NetTRUST, we update the CTCS private-shares in order to defend against attacks of the *mobile adversaries*. This category of attacks was, initially, studied in [31] to characterize the adversaries which, temporarily, compromise a server and pass to another victim server (for example, a virus injected into the network). An adversary is able to compromise k MCA servers in a period time, called *window of vulnerability* [12]. If the adversary succeeds in collecting k private-shares from compromised MCA servers, he will be able to reconstruct the certification service private-key and to sign wrong certificates in the system. The CCA servers recreate and redistribute new private-shares to MCA servers at defined time periods. The new private-shares also constitute a (k, n) threshold scheme. Therefore, MCA servers must remove the old private-shares and use the new ones in the CTCS. The new private-shares must be independent of the olds. The adversary should not be able to combine the new private-shares and the old ones to discover the CTCS private-key. Thus, the adversary is opposed to compromise k MCA servers in the time relating to the update period.

[2]The CTCS public-key is known for all users in the network.

[3]For example, no access to the wired part of the network.

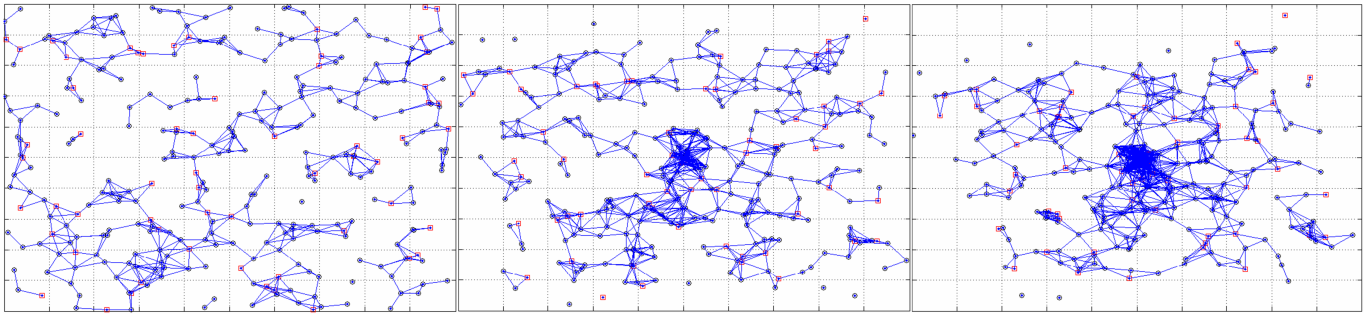

Fig. 9. Change of Topology in the Ad−hoc Network Under the Mobility Model Towards the Center: {(a)→(b)→(c)}. Nodes with Circles, MCA Servers with Squares, and Communication Links with Lines.

IV. Simulations

In this section, we evaluate the performance of NetTRUST with respect to the rate of successful certificates, under the failures of servers and the network partitioning.

A. Parameters and Assumptions

In our simulations, we have used the MATLAB environment.[4] We simulate a mixed network as an ad-hoc network with 100 nodes connected to a wired network through access points. Each node has a nominal range α and move on a square area of 1km^2. Our simulator estimates if a radio link exists between two nodes according to the distance that separates them. We used a particular model of mobility, inherited from the model of C. Zhang et al. [13], where nodes move in the area toward the center of the surface (cf. Figure-9). The initial nodes positions are random on the surface, and their movements' speed interval is defined in [0ms,20ms]. We assume that nodes have the same hardware characteristics and processing capabilities, and are configured by wireless communication interfaces of 22Mbps transmission rate. Our evaluation of NetTRUST focuses on the measurement of the success rate of certification requests. Every node which receives k or more partial certificates is counted as a successful certification request and the related success rate is defined as:

$$\frac{\text{Number of Successful Certification Request}}{\text{Number of Total Certification Request}}$$

In our simulations, the CTCS contains 50 MCA servers, which are assumed to be predefined. The protocols of NetTRUST are omitted to simplify the analysis of the results. In the carried out simulation, we consider a duration of 3600 seconds. We assume that certificate queries arrive following a Poisson law with an inter-arrival between queries of 10 seconds.

First, we investigate the impact of the network partitioning on the performance of NetTRUST. We evaluate this metric following two cases: lose access to the wired network and nodes inter-disconnections. Then, we study the impact of the MCA servers' failures. We, also, consider that failures arrive at the MCA servers according to the Poisson law with an inter-failures average ψ and remain broken down during ξ.

[4]The name MATLAB is an abbreviation of MATrix LABoratory. MATLAB is an interactive, matrix-based system for scientific and engineering calculations. Designated to solve complex numerical problems.

B. Impact of Network Partitioning

The CCA servers are the central certification authorities, and they are fixed on the wired part of the network. Therefore, we assume that energy is always available for CCA servers. The role of CCA servers is to deliver certificates for users, and to delegate the MCA servers. The users in the ad-hoc part of the network can request CCA servers directly through the access points, if they are accessible. Otherwise, users must request the MCA servers.

Firstly, we were interested in studying the performance of NetTRUST with and without access to the wired network. We simulated two models, the first with CTCS using a $(k, 50)$ threshold cryptography scheme. The second with CTCS using a $(k, 50)$ threshold cryptography scheme and DTCS using a $(1, 5)$ threshold cryptography, where CCA servers can be accessible or not through access points. We varied the value of k and we observed the average rate of successful certificates in the two cases.

Figure-11 shows that with access to the wired network, the successful certificates ratio is obviously more important than the successful certificates ratio in the case the wired network is not accessible. The users having access to CCA servers can request them without broadcasting a certification request to the MCA servers. When we increase the value of k, the rate of successful certificates decreases. When k is large; the users are required to request a large number of partial certificates. If k is very large, the users risk not being able to recover k partial certificates and this is due to the possible unavailability of MCA servers. Also, if k is reduced, the users will request a narrow number of MCA servers, and they will have many chances to have more certification responses.

We define two metrics: certification service $availability$, and certification service $robustness$. The service availability is interpreted as the capacity to respond to the users' request. The robustness is interpreted as the capacity of NetTRUST

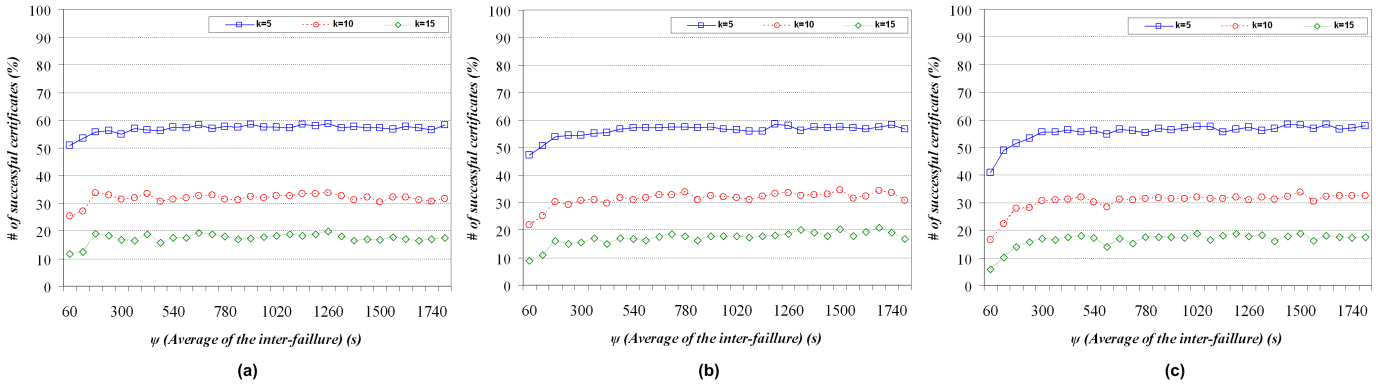

Fig. 10. Impact of ψ (Average of the Inter−Failures) on # of Successful Certificates for (a) $\xi = 900$s, (b) $\xi = 1800$s, (c) $\xi = 2700$s. $t = 1$, $m = 5$, $\alpha = 100$m.

to resist against adversaries, which collect private-shares from compromised servers.

Consider the case when NetTRUST operates on the ad-hoc part of the network, which is completely disconnected from the wired part of the network. If k is small, it will be relatively less difficult for adversaries to compromise the CTCS private-shares. In this case, the robustness degree decreases, but the service availability becomes stronger. If k is great, the adversary will be required to compromise much more MCA servers to be able to compromise all the system. Thus, the robustness increases. However, the users must request many more MCA servers to be able to satisfy their requests, then the service availability decreases.

When NetTRUST operates on mixed networks, we can freely increase the value of k. Increasing the value of k will increase the robustness degree of the service, at the same time the service availability remains stable at an acceptable performance (65% as shown in Figure-11). Therefore, NetTRUST ensures the two metrics: $availability$ and $robustness$.

$$(k) \uparrow \Rightarrow (availability) \downarrow \wedge (robustness) \uparrow$$
$$(k) \downarrow \Rightarrow (availability) \uparrow \wedge (robustness) \downarrow$$

Then, we were interested in studying the impact of the nodes disconnections on the performance of NetTRUST. Thus, we set the DCTS scheme coordinates to $(1, 5)$, and vary the nominal range α, as well as threshold parameter k of the CTCS scheme. The Figure-12 represents the impact of α on the average rate of successful certificates. For reduced α, the performances of NetTRUST are bad. Starting from nominal range α=100m the performances become very interesting. So, practically NetTRUST is more powerful in the application areas where nodes are configured by wireless interfaces beyond 100m. This is not a strong assumption, since most of the actual technologies provide communication ranges greater than 100m.

C. Impact of MCA Servers' Failures

We first set the nominal range α at 100m and the DCTS scheme coordinates to $(1, 5)$, and vary the average of the inter-failures ψ of MCA servers, as well as the threshold parameter

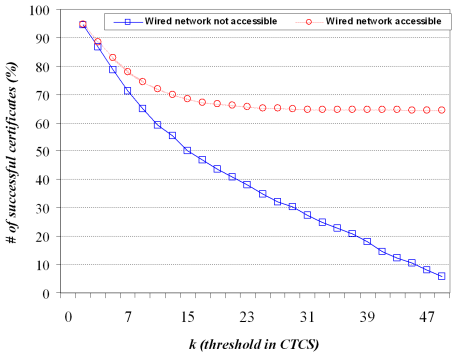

Fig. 11. Evaluation of Service Under k (Threshold in CTCS) with/without Access on the Wired Network on # of the Successful Certificates. $t = 1$, $m = 5$, $\alpha = 100$m.

k of the CTCS scheme. The figure-10 (a), (b), and (c) show the impact of ψ on the average rate of successful certificates for, respectively, $\xi = 900$ seconds, $\xi = 1800$ seconds, and $\xi = 2700$ seconds (ξ is the average duration of failure).

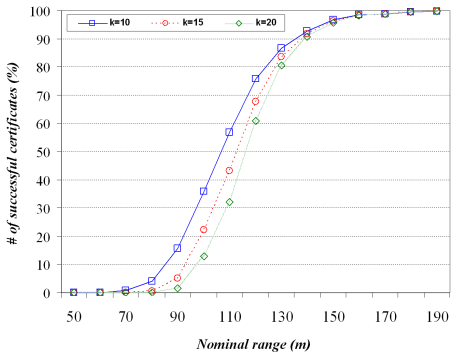

Fig. 12. Impact of α (Nominal Range) on # of Successful Certificates. $t = 1$, $m = 5$.

During the variation of ψ the performances of NetTRUST remained, approximately, stable. Also, the performances are, approximately, similar during the variation of the period ξ of MCA servers failures. This means that NetTRUST resists against the $density$ and the $durability$ of MCA servers failures. Therefore, NetTRUST is very well adapted for applications which undergo, frequently, machines' failures. For example, a military application during wars period.

V. Conclusion

In this paper, we have focused on the trust models and infrastructures in both ad-hoc and wired networks. We have presented taxonomy of the related models, where we have classified models in three principal categories: centralized models, partially distributed models, and completely distributed models. We have then proposed NetTRUST, for mixed networks architectures. NetTRUST uses two particular certification authorities, that ensure $\mathrm{X}509v3$ certificates management: the central certification authorities (CCA) are tied to the portions of the network having a pre-existent communication infrastructure (such as wired networks, cellular networks, etc), and mobile certification authorities (MCA) which are on the ad-hoc portion of the network. The MCA servers emulate the certification authority role using a (k, n) threshold cryptography scheme, and the CCA servers delegate the role of certification to the MCA servers using a (t, m) scheme of threshold cryptography. This solution is decentralized and partially distributed, supports the nodes mobility and the failure of, up to $n-k$, among n MCA servers. The simulation results and the performance evaluation prove the adequacy of this solution to mixed networks architectures.

The contributions of NetTRUST include: $1)$ the network architecture framework in which operates, $2)$ the hierarchical delegation between central and mobile certification authorities, $3)$ double threshold scheme: one to distribute the certification service, and the second to distribute the delegation service, and $4)$ threshold certification management based on the $\mathrm{X}509v3$ certificate format.

Acknowledgments

I would like to thank Sihem Yahiaoui, Fouzi Semcheddine, and Salah Guesmia for their helps and encouragements. Also, authors are grateful to Stephen Marsh for his help in reviewing the final draft.

References

[1] S. Raghani, D. Toshniwal, and R. Joshi. Dynamic Support for Distributed Certification Authority in Mobile Ad Hoc Networks. *In IEEE International Conference on Hybrid Information Technology*, 2006.

[2] J. Luo, J. Hubaux, and P. Eugster. DICTATE: Distributed Certification Authority with Probabilistic Freshness for Ad hoc Networks. *IEEE Transactions on Dependable and Secure Computing*, 2005.

[3] B. Wu, J. Wu, E. Fernandez, and S. Magliveras. Secure and Efficient Key Management in Mobile Ad Hoc Networks. *In Proceedings of the* 19^{th} *IEEE International Parallel and Distributed Processing Symposium*, 2005.

[4] A. Pirzada and C. McDonald. Kerberos Assisted Authentication in Mobile Ad-hoc Networks. *In Proceedings of* 27^{th} *Australasian Computer Science Conference*, 2004.

[5] J. Luo, P. Eugster, and J. Hubaux. PILOT: Probabilistic Lightweight group communication system for Mobile Ad hoc Networks. *IEEE Transactions on Mobile Computing*, 2004.

[6] Q. He, D. Wu, and P. Khosla. SORI: A secure and objective reputation-based incentive scheme for ad hoc networks. *In IEEE Wireless Communications and Networking Conference*, 2004.

[7] P. Ratanchandani and R. Kravets. A Hybrid Approach to Internet Connectivity for Mobile Ad Hoc Networks. *In Wireless Communications and Networking Conference IEEE*, 2003.

[8] S. Yi and R. Kravets. MOCA: Mobile Certificate Authority for Wireless Ad hoc Networks. *In Proceedings of* 2^{nd} *Annual PKI Research Workshop*, 2003.

[9] M. Hwang, E. Lu, and I. Lin. A Practical (t, n) Threshold Proxy Signature Scheme Based on the RSA Cryptosystem. *IEEE Transactions on Knowledge and Data Engineering*, 2003.

[10] S. Capkun, L. Buttyan, and J. Hubaux. Self-ogranized public key management for mobile Ad hoc networks. *IEEE Transactions on Mobile Computing*, 2003.

[11] C. Adams and S. Lloyd. Understanding PKI: Concepts, Standards, and Deployment Considerations. 2^{nd} *Edition Addison Wesley*, 2002.

[12] L. Zhou, F. Schneider, and R. Renesse. COCA: A Secure Distributed Online Certification Authority. *ACM Transactions Computing Systems*, 2002.

[13] C. Zhang, B. DeCleene, J. Kurose, and D. Towsley. Comparison of Inter-Area Rekeying Algorithms for SecureWireless Group Communications. *Elsevier Science Publishers*, 2002.

[14] Y. Sun, E. Belding-Royer, and C. Perkins. Internet Connectivity for Ad Hoc Mobile Networks. *International Journal of Wireless Information Networks*, 2002.

[15] M. Ma and C. Meinel. A Proposal for Trust Models: Independent Trust Intermediary Service. *International Association for Development of the Information Society*, 2002.

[16] S. Buchegger and J. Boudec. Performance Analysis of the CONFIDANT Protocol: Cooperation of Nodes-Fairness In Dynamic Ad-hoc NeTworks. *In Proceedings of IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing*, 2002.

[17] P. Michiardi and R. Molva. CORE: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. 6^{th} *IFIP Conference on Security Communications and Multimedia*, 2002.

[18] ITU−T Recommendation. Public-Key and Attribute Certificate Frameworks. 4^{th} *Edition*, 2001.

[19] D. Malkhi, M. Reiter, and A. Wool. Probabilistic Quorum Systems. *Information and Computation*, 2001.

[20] 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 International Conference on Network Protocols (ICNP'*01*)*, 2001.

[21] H. Luo and S. Lu. Ubiquitous and Robust Authentication Services for Ad hoc Wireless Networks. *Technical Report, UCLA Computer Science*, 2000.

[22] S. Marti, T. Giuli, and K. Lai. Mitigating routing misbehavior in mobile ad hoc networks. *In Proceedings of MOBICOM'*00, 2000.

[23] R. Perlman. An Overview of PKI Trusts Models. *IEEE Network*, 1999.

[24] S. Chokhani and W. Ford. Internet X509 PKI Certificate Policy and Certification Practices Framework. *RFC*−2527, 1999.

[25] L. Zhou and Z. Haas. Securing Adhoc Networks. *IEEE Network*, 1999.

[26] K. Birman, M. Hayden, and O. Ozkasap. Bimodal Multicast. *ACM Transactions Computer Systems*, 1999.

[27] A. Abdulrahman. The PGP Trust Model. *The Journal of Electronic Commerce*, 1997.

[28] A. Abdulrahman and S. Hailes. A Distributed Trust Model. *In Proceedings* 97 *New Security Paradigms*, 1997.

[29] C. Neuman and T. Ts'o. Kerberos: an Authentication Service for Computer Networks. *IEEE Communications*, 1994.

[30] J. Kohl and B. Neuman. The Kerberos Network Authentication Service Version 5. *RFC*−1510, 1991.

[31] R. Ostrovsky and M. Yung. How to Withstand Mobile Virus Attacks. *In Proceedings of the* 10^{th} *ACM Symposium on Principles of Distributed Computing*, 1991.

[32] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digitial signatures and public key cryptosystems. *Communication of the ACM*, 1978.

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