DISTANT: Distributed Trusted Authority-based Key Management for Beyond 5G Wireless Mobile Small Cells

Marcus de Ree[a,b,*], Georgios Mantas[a,c], Jonathan Rodriguez[a,b], Ifiok E. Otung[b], Christos Verikoukis[d]

[a] *Instituto de Telecomunicações, Campus Universitario, Aveiro, Portugal*
[b] *Faculty of Computing, Engineering and Science, University of South Wales, Pontypridd, UK*
[c] *Faculty of Engineering and Science, University of Greenwich, London, UK*
[d] *Centre Tecnològic Telecomunicacions de Catalunya, Barcelona, Spain*

Abstract

The 5G mobile network is embracing new technologies to keep providing network subscribers with a high Quality of Service (QoS). However, this has become increasingly difficult in the urban landscape as more devices are being connected and each device is requesting increasing amounts of data. Network operators rely on the small cell technology to maintain coverage and service for its subscribers, but this technology is incapable of mitigating the increasing workload on the network infrastructure and preventing the associated network delays. The next logical step is to cover the urban landscape with mobile small cells, since these take advantage of the dynamic network topology and optimizes network services in a cost-effective fashion while taking advantage of the high device density. However, the introduction of mobile small cells raises various security challenges. Cryptographic solutions are capable of solving these as long as they are supported by an appropriate key management scheme. In this article, we propose DISTANT: a DIStributed Trusted Authority-based key managemeNT scheme. This key management scheme is specifically designed to provide security in a network which takes advantage of the mobile small cell technology. The scheme relies on threshold secret sharing to decentralize trust and utilizes the self-generated certificates paradigm. Through an extensive security analysis and communication overhead evaluation, we conclude that our design provides an improved level of security and has a low communication overhead compared to previous works.

Keywords: Beyond 5G Security, Decentralized System, D2D Communication, Key Management, Mobile Small Cell

1. Introduction

The mobile network entered the 5G era, bringing emerging networking technologies to handle the immense growth of the mobile network. The mobile network had approximately 6.5 billion connected devices, requesting 0.9 exabytes of mobile data per month by 2012 [1]. It is forecasted that the number of connected devices will grow to 12.3 billion by 2022, requesting 77 exabytes of mobile data per month [2]. This surge puts a lot of pressure on the mobile network which has to share its resources and will undoubtedly lead to a reduction in the delivered QoS.

To address these challenges, new technologies are emerging to create the next generation 5G network. One of these is the small cell technology [3]. This technology is based on the femto-cell paradigm and is realized by installing small and low powered radio access nodes to provide coverage in densely populated areas. These nodes act as mini base stations and provide significant benefits such as reduced power consumption, increased data rates and reduced latency. However, small cell technology does not reduce the workload of the network infrastructure which can become a bottleneck in densely populated areas. The EU-funded H2020-MSCA project "SECRET" introduced a system model which utilizes so-called mobile small cells [4]. These mobile small cells form a wireless network of small cells and exists entirely of mobile devices. Data transmissions are established through device-to-device (D2D) communications (and multiple hops if necessary) without having to rely on the existing network infrastructure. This system model is particularly suitable for an urban environment since the high network density guarantees a communications path between network subscribers. Technologies such as network coding could be utilized to provide significant benefits to networks in terms of bandwidth, energy consumption, delay and robustness to packet losses [5]. This system model therefore provides additional benefits compared to ordinary small cells and could potentially function alongside the 5G mobile network, providing a high level of QoS to communicating network subscribers which are within relative close proximity while offloading the network infrastructure.

However, mobile small cells raise significant challenges in terms of security and privacy. Cryptographic security solutions (e.g., encryption and integrity schemes) are capable of solving these as long as they are supported by a key management scheme [6]. A key management scheme dictates how cryptographic keys are organized within a network such that they can be used effectively to secure communication between any set

*Corresponding author.
Email address: mderee@av.it.pt (Marcus de Ree)

of users. Generally, key management schemes rely on some form of an online centralized trusted third party (TTP) to provide trust and security. This TTP is considered trustworthy and secure by every user inside the network. However, not a single network entity should distribute cryptographic keying material in an online fashion as they are all vulnerable to denial-of-service (DoS) attacks or physical compromise. Enabling secure multi-hop D2D connectivity using mobile small cells therefore requires network independent security solutions. Security must be guaranteed by means of a key management scheme which decentralizes trust.

In this article, we present the DISTANT (DIStributed Trusted Authority-based key managemeNT) scheme and is based on our previous work [7]. The core feature of this scheme is the combination of threshold secret sharing [8] with self-generated certificates [9, 10]. Threshold secret sharing has trust distributing capabilities which allow key management services to be provided in a decentralized manner. Furthermore, verifiable secret sharing (VSS) [11, 12] and proactive secret sharing (PSS) [13, 14] provides robustness against malicious adversaries. The network nodes, defined as a mobile device in possession of a network subscriber, are provided with proxy keys that allows them to issue and sign certificates for themselves as if they were issued by the TTP. These proxy keys enable non-interactive certificate updates and reduce the communication overhead. DISTANT's primary focus has been on minimizing the communications overhead as this restriction remains relevant over time whereas the computational and memory storage overheads become less of an issue as technology improves. The proposed protocols are evaluated analytically for its security strength and its performance. Results show that the protocols are mathematically sound and that DISTANT provides a higher level of security and in many cases a reduction in communication overheads compared to related works. To sum up our main contributions:

- We designed a novel and decentralized key management scheme that is capable of supporting cryptographic security solutions (e.g., encryption and integrity schemes) to secure Beyond 5G wireless mobile small cells in dense, urban environments.
- The key management scheme is the first of its kind to be resistant to all the relevant malicious attacks, specified in the adversarial model.
- We provided guidelines for selecting an appropriate security threshold, based on the size and node density of the network through performed simulations.
- We evaluated and compared the performance of our novel key management scheme with related key management schemes and found the design to be either equally or more efficient from a communication overhead perspective.

The remainder of this article is organized as follows. A review of related works is provided in section 2. The details of the system model is covered in section 3. Sections 4 and 5 cover the adversarial model and security objectives, respectively. Details of the proposed key management scheme is covered in section 6, followed by the security analysis in section 7 and the performance evaluation in section 8. Finally, our conclusions are covered in section 9.

2. Related Work

Research into decentralized and self-organized key management solutions took off around the turn of the century when the concept of the mobile ad hoc network (MANET) was introduced. Providing security in such networks has been a challenge since ordinary networks generally have access to a secure and reliable TTP. The lack of such a TTP in MANETs required security researchers to come up with novel key management schemes to provide secure multi-hop D2D communications. Our system model is similar to that of a MANET (with a few exception) and are therefore inspired by those key management solutions. The recent survey [6] identified that the fully distributed TTP (FD-TTP)-based key management solution, originally proposed for MANETs, is the most suitable candidate to establish secure communication in our system model.

The FD-TTP-based key management solution was proposed by Luo et al. [15, 16]. Traditionally, a centralized TTP is in possession of a master key pair that is used to provide key management services (e.g., issue certificates or generate and distribute private keys). In this solution, the master private key is split into shares using threshold secret sharing techniques [8]. These shares are then distributed to a proper subset of nodes, called servers, such that a threshold amount of them can collaboratively provide the key management service during network operation. However, the use of a distributed TTP comes with a variety of security challenges which must be addressed. The incorporation of VSS [11, 12] allows joining nodes to verify that an honest key management service is provided. Therefore, malicious servers can be detected. The incorporation of PSS [13, 14] allows servers to periodically update their shares, protecting the secrecy of the master private key in long-lasting networks. Finally, a malicious server that manages to compromise enough shares of the master private key, enabling its reconstruction, is capable of impersonating the FD-TTP. The trust level [17] of the FD-TTP describes the malicious capabilities in case of compromise. The malicious capabilities are most severe at level 1 and the least severe (and even detectable) at level 3. A key management scheme with an FD-TTP that reaches trust level 3 is therefore the most desirable, since it provides an additional layer of security. More details about the relevant attacks and security objectives are covered in sections 4 and 5, respectively. Unfortunately, neither of the previously proposed schemes that follow the FD-TTP-based key management solution satisfies all the security objectives. The security drawbacks of these schemes are summarized in Table 1.

The initial proposal by Luo et al. [15, 16] is based on traditional public key infrastructure (PKI). In this cryptographic infrastructure, every network node generates their own public-private key pair and requests the FD-TTP to certify its public key. Upon receiving a threshold amount of partially signed certificates, the network node combines these to obtain its issued

Table 1: Security drawbacks of FD-TTP-based key management schemes.

FD-TTP-based Key Management Scheme	Trust Level of the FD-TTP	Verifiable Secret Sharing	Proactive Secret Sharing
Luo et al. [15, 16]	3	✓[1]	✓
Deng et al. [18]	1	✗	✗
da Silva et al. [19]	1	✗	✗
Zhang et al. [20]	2	✗	✗
Li et al. [21]	2	✓[2]	✓
Gharib et al. [22]	2	✗	✗
Lai et al. [23]	3	✗	✗

[1]Verifiability is limited to the combined key management service. The inability to verify partial key management services prevents the detection of malicious servers.
[2]Verifiability is limited to the distributed key establishment protocol. Their distributed secret share establishment protocol and the secret share updating protocol lack verifiability.

certificate. Unfortunately, it was demonstrated in [24] that network nodes are unable to verify whether the obtained partial certificates and partial secret shares are correct. This allows a malicious server to provide a faulty key management service without being detected.

Deng et al. [18] and da Silva et al. [19] proposed FD-TTP-based key management schemes that are based on identity-based public key cryptography (PKC). In this cryptographic infrastructure, the node's identity (e.g., MAC address or phone number) is used as the user's public key. A node's private key can be computed using the master private key; thus, every node must request the FD-TTP for pieces of its private key. This cryptographic infrastructure achieves only Girault's trust level 1 [17, 25]. Therefore, a compromised FD-TTP gains tremendous power to launch malicious attacks without being detected. It has been suggested that schemes based on identity-based PKC are more suitable in small and closed networks with limited security requirements due to this drawback [20, 21].

Zhang et al. [20], Li et al. [21], Gharib et al. [22] and Lai et al. [23] proposed FD-TTP-based key management schemes that are based on certificateless PKC [26]. This cryptographic infrastructure is a hybrid between traditional PKI and identity-based PKC. A network node essentially combines the self-generated public-private key pair with an identity-based public-private key pair. The self-generated public key and the node's identity are combined into the node's public key and the self-generated private key and the identity-based partial private key (obtained from the FD-TTP) are combined into the node's private key. Al-Riyami et al. [26] showed that the TTP can reach either trust level 2 or trust level 3, depending on the key generation technique. By inspection, we found that [20, 21, 22] only reach FD-TTP trust level 2, whereas the scheme by Lai et al. [23] reaches trust level 3. This means that only their scheme is capable of detecting malicious activities in case the network becomes compromised. Unfortunately, their key management scheme does not incorporate VSS[1] or PSS, making their design still vulnerable against certain malicious adversaries.

3. System Model

3.1. System Description

The system model that consists of virtual mobile small cells for the next generation mobile network, was introduced by the H2020-MSCA project "SECRET" [4] and is illustrated in Figure 1. In this model, the mobile network (consisting of macro cells) is virtually partitioned into a network (or cloud) of mobile small cells, containing heterogeneous mobile devices such as smartphones and laptops. Each mobile small cell has an approximate radius of 50 to 100 meters and is controlled and maintained by a hotspot. This hotspot is a heterogeneous mobile device that is selected to become the local radio manager to control and maintain the cluster. In addition, each hotspot is controlled by a centralized software-defined controller. Through cooperation these hotspots form a wireless network that has several gateways to the mobile network using intelligent high-speed connections. Data traffic between devices is established through multi-hop D2D communications. This system model can function alongside the 5G mobile network, providing a high level of QoS to communicating network subscribers which are within relative close proximity, while offloading the network infrastructure. The network subscribers can take advantage of communicating through these mobile small cells in the one-on-one setting as well as the group setting, applied to voice calling, video calling, text messaging and data exchange (e.g., exchange of data messages, pictures, videos, data related to a multi-player game, etc.).

The densification of the urban landscape by means of mobile small cells provides opportunities for both network operators and network subscribers. Network subscribers are provided with an increase in data rates and a reduction in power consumption and latency, while network operators benefit from a reduction in signal interference, network offloading and network operating costs. However, many of these advantages can be credited to the introduction of ordinary small cells. Since the strength of a radio signal diminishes with the square of the distance, replacing large transmissions to and from the base station by multiple shorter transmissions provide significant energy savings and reduce the amount of interfering radio signals. The energy savings could then be invested towards enabling higher data rate transmissions. Furthermore, the physical propagation distance is significantly reduced when a source node and a destination node are within relative close proximity and thus reduces latency.

Nevertheless, mobile small cells provide additional advantages. They can be set up on-the-fly, based on demand, at any place, at any time, using existing mobile devices. These mobile devices can propagate data through the network using multi-hop D2D communications and enables network offloading. Densely

[1]The use of verifiable secret sharing is mentioned; however, it is not incorporated in any protocol.

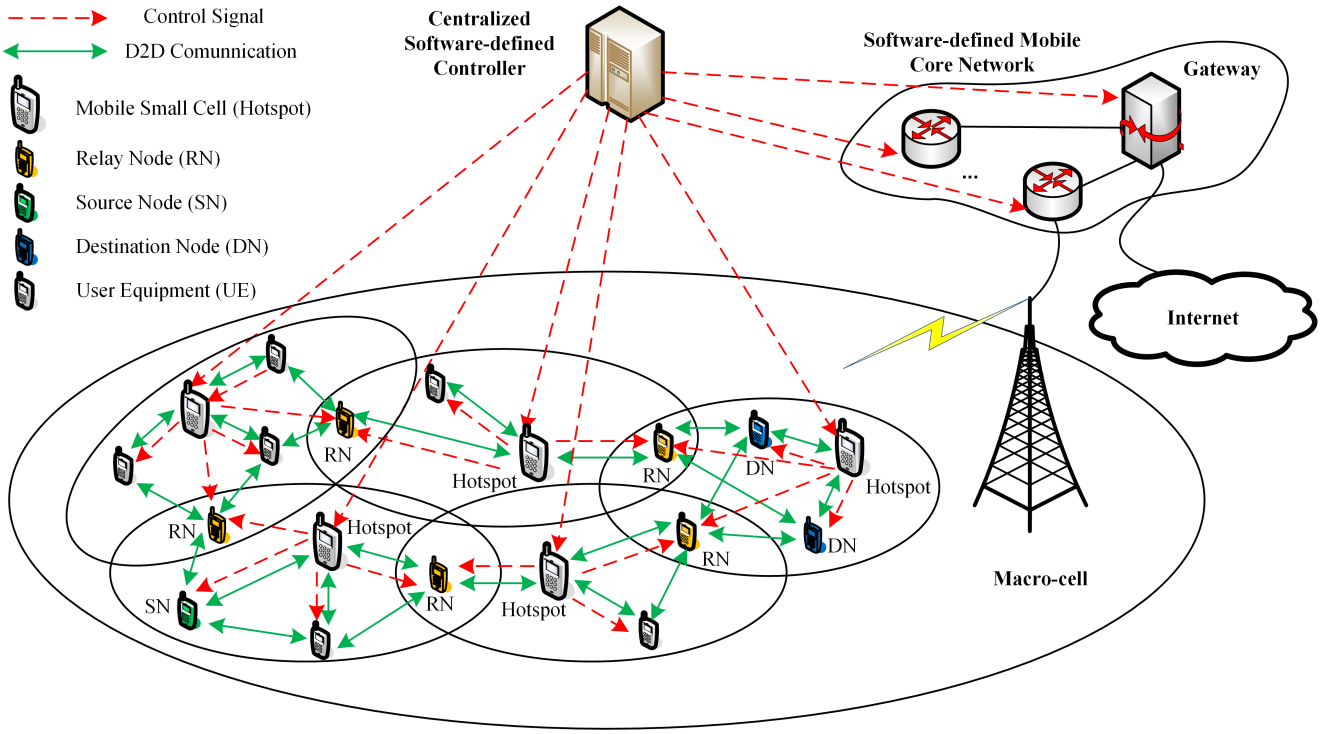

Figure 1: The system model as introduced by H2020-MSCA project "SECRET" [27].

populated urban environments would benefit greatly from this as network congestion in these environments are most prolific. Furthermore, since the mobile small cells are formed from existing mobile devices, network operators do not have to install or maintain additional network infrastructure which reduces the network operating costs. Moreover, the dynamic network topology supports time and space varying traffic [28]. Finally, network coding could be utilized to provide significant benefits to networks in terms of bandwidth, energy consumption, delay and robustness to packet losses [5].

Recently, the mobile telephony standardization organization 3GPP introduced the concept of "Indirect 3GPP Communication" to extend coverage. This is defined as the signaling and communication between a mobile device and a 3GPP network via one or more relay nodes in [29]. This concept covers the utilization of multi-hop D2D communications as covered in mobile small cells, demonstrating its technological relevance.

3.2. Assumptions

For our key management scheme, we consider a network of mobile small cells which cover an urban environment. This urban environment has the highest node density in the center and has a gradually decreasing node density as we distance ourselves from the center. The network contains n network nodes and the size of the network is defined as the area within the urban environment in which every network node generally has at least the threshold t amount of neighbors within its direct transmission range. The network topology is dynamic, network nodes can move freely inside the network and nodes may join or leave the network at any time. We assume the existence of a TTP (e.g., a network operator or a collaborative effort from multiple network operators) during network initialization to bootstrap an initial set of nodes. We assume that the transmission range of each node is equal and that they are capable of sending unicast, multicast or broadcast messages. We define a unicast message as a message which is cryptographically secured (i.e., by means of encryption) by the sender and only one receiver has the corresponding cryptographic keying material to extract the information from the message. Similarly, we define a multicast message as a message which is cryptographically secured by the sender and a set of multiple receivers have corresponding cryptographic keying material to extract the information from the message. We define a broadcast message as a message which is sent in plaintext such that every network node within the transmission range of the sender receives this message. These messages are invulnerable to malicious message modification attacks when both the sender and receiver are within each other's transmission range. We assume that each of the protocols which involves communication between multiple nodes remain within each other's transmission range during the execution of the protocol. Finally, we assume that network nodes periodically send beacon messages to informs nearby and incoming nodes that they are within the boundaries of the network.

4. Adversarial Model

In the FD-TTP-based key management solution, that relies on the distribution of trust through secret sharing techniques, the most important aspect of security is the continued secrecy

of the master private key. We discuss, underneath, two types of attacks that are related to the establishment and maintenance of a secure and trustworthy key management service and one important and inherent characteristic of public key cryptographic infrastructures and their impact on providing security for a FD-TTP-based key management scheme.

Disruptive Adversary Attack. In the disruptive adversary attack, a malicious server provides a false key management service to a requesting node. This false key management service can be categorized in the following two ways:

1. *False partial secret share*
   The disruptive adversary could send a false partial secret share to a joining node, leading to the creation of an incorrect secret share. Consequently, this incorrect secret share will cause a well-behaving server to unknowingly provide false key management services to joining nodes in the future.
2. *False partial keying material*
   The disruptive adversary could also send false partial keying material (e.g., a false partially signed certificate or a false partial private key) to a requesting node, leading to the creation of incorrect keying material. This is then followed by the inability to establish a secure communication channel with other network nodes. Therefore, a disruptive adversary can have crippling effects on the key management service and prevents nodes from establishing secure communication.

Mobile Adversary Attack. In the mobile adversary attack [30], a malicious adversary moves dynamically through the network and compromises network nodes, one at a time, with the goal to extract and collect a threshold amount of shares of the master private key. If the mobile adversary is successful, it is capable of reconstructing the master private key. This allows the adversary to impersonate a distributed TTP and launch a variety of malicious attacks. The severity of these attacks depends on the public key cryptographic infrastructure of the key management scheme, as this defines the trust level of the distributed TTP.

Trust Level of the Distributed TTP. Girault [17] found that public key cryptographic infrastructures have a variety of trust levels. He defined a hierarchy of three trust levels as follows:

1. The TTP knows (or can easily compute) a node's private key and can launch identity impersonation attacks without being detected.
2. The TTP does not know (and cannot easily compute) a node's private key but is still able to launch identity impersonation attacks without being detected.
3. The TTP does not know (and cannot easily compute) a node's private key nor is it able to launch identity impersonation attacks without being detected.

In the event that the master private key is exposed, the adversary in possession of the master private key is capable of impersonating a malicious TTP. These malicious capabilities depend on the trust level of the distributed TTP. This characteristic is important in the design of a key management scheme as it can provide an additional layer of security.

5. Security Objectives

To prevent the attacks described in the adversarial model, the following security objectives must be achieved.

The incorporation of Verifiable Secret Sharing. To mitigate the disruptive adversary attack, a requesting node must be capable of verifying the correctness of the provided key management service. This becomes possible by incorporating verifiable secret sharing [11, 12] into the key management design. Furthermore, the detected misbehaving servers can be removed from the network. Incorporating verifiable secret sharing should therefore discourage servers from providing a false key management service.

The incorporation of Proactive Secret Sharing. The secrecy of the master private key must be maintained during the entire network lifetime. Since the mobile network has a long lifetime, a mobile adversary has an extensive period of time to make its attack successful. To limit the window of opportunity, we require network nodes to be proactive when it comes to maintaining the secrecy of the master private key. Therefore, network users should periodically update their secret shares [13, 14] such that a mobile adversary is incapable of collecting a threshold amount of secret shares in between two share updating phases.

The Distributed TTP must reach Trust Level 3. Having a FD-TTP which reaches trust level 3 provides the highest level of security since it allows the detection of network compromise. This gives network operator(s) the ability to reboot the network with enhanced security parameters, such as a higher security threshold or a reduced interval between share updating phases. The limited payoff should discourage adversaries from attempting to compromise the network.

6. The DISTANT Scheme

The DISTANT scheme, schematically illustrated in Figure 2, has three phases. The first phase is the network initialization phase. In this phase, we rely on a TTP (such as a network operator or a collaborative effort from multiple network operators) to execute the network setup algorithm and inject trust into the network by bootstrapping an initial set of nodes. These initial nodes are provided the public network parameters along with their initial keying material (i.e., a share of the master private key, a personalized proxy key pair and the public witness values). After at least a threshold amount of nodes are initialized, the TTP leaves the network and the network becomes operational in a self-organized manner.

The second phase is the operational phase. In this phase, network nodes can use their proxy key pair to issue and update their own self-generated certificate, as if it was issued directly by the TTP, in a non-interactive fashion. Any arbitrary set of

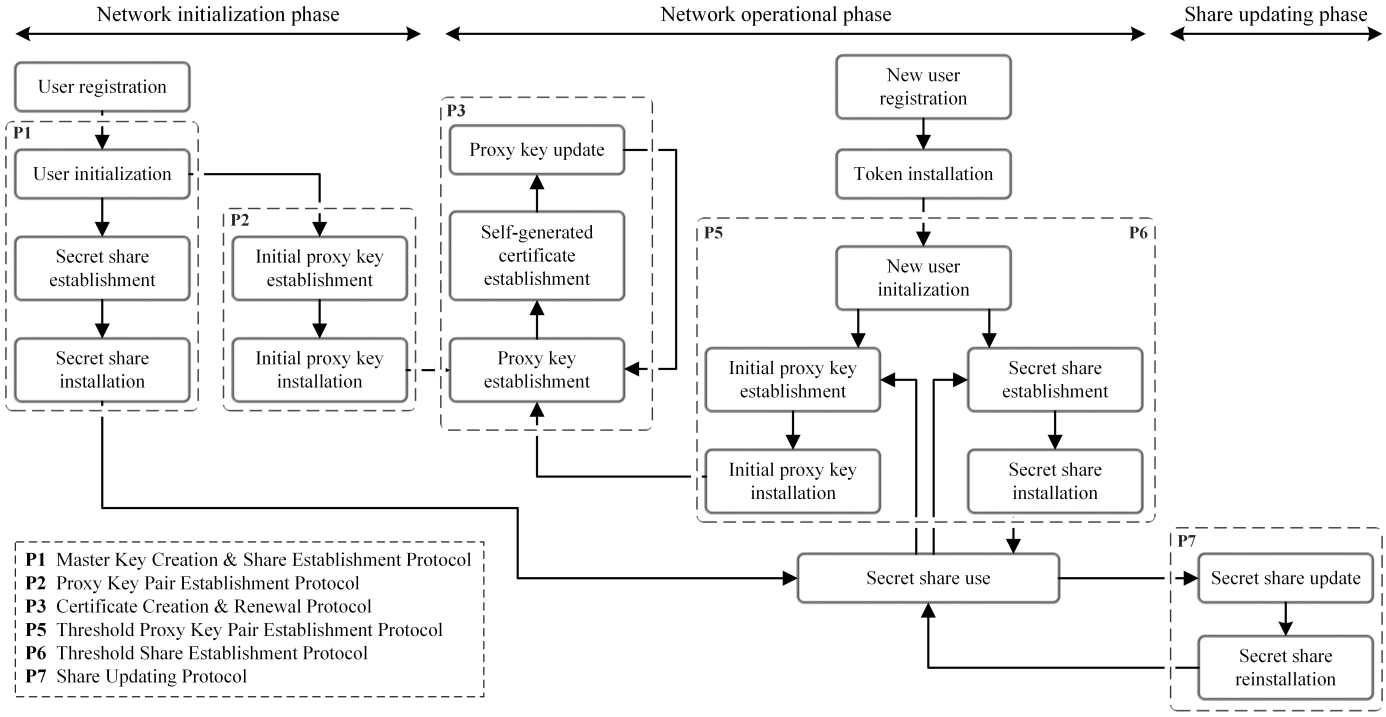

Figure 2: The schematic design of DISTANT with protocol localization.

network nodes can exchange their self-generated certificates, verify their authenticity, and establish a secure communication channel. It is also possible for new nodes to join the network during the operational phase. Joining nodes can broadcast a joining request to obtain its initial keying material. The network nodes that are within transmission range of the joining node and receive the joining request, can use their share of the master private key to provide partial proxy key pairs and partial shares of the master private key to the joining node. The joining node can combine these pieces to establish its own proxy key pair and share of the master private key.

The third phase is the share updating phase and it is triggered periodically. In this phase, every network node updates its share of the master private key. The obtained and updated share will be independent from its previous shares, effectively preventing a mobile adversary from successfully collecting a threshold number of shares in between share updating phases. This provides long-term security for our network.

6.1. Network Initialization Phase

The network is initialized by a TTP and a cluster of t initial nodes. The TTP could be a network operator or a collaborative effort from multiple network operators. We assume the existence of a secure channel between the TTP and each initial node during this phase. The following algorithm and protocols are executed during this phase:

1. *Network Setup*: The TTP executes the network setup algorithm to define public and private network parameters.
2. *Centralized Secret Share Establishment*: The TTP computes and provides each initial node with a personalized secret share of the master private key.
3. *Centralized Proxy Key Pair Establishment*: The TTP initiates an interactive protocol with each initial node to establish their initial proxy key pair.

6.1.1. Network Setup

The network setup algorithm is executed by the TTP. The algorithm generates, selects and defines public network parameters (i.e., primes p and q, generator g, security threshold t, master pulic key MPK and the set of witness values ω) and public network functions (i.e., hashes h_1, h_2, h_3, signature scheme SIG, public encryption scheme ENC, message authentication code MAC and symmetric encryption cipher CIP). These network parameters and functions are published such that every node knows how to perform its mathematical operations. Furthermore, the TTP generates a master polynomial $f(x)$ that defines the master private key MSK and allows the TTP to compute the shares of the master private key for the initial set of nodes. The technical details are described in Algorithm 1.

6.1.2. Centralized Secret Share Establishment

In this protocol, the TTP bootstraps the network and establishes the distributed TTP by providing an initial node with a share of the master private key. The TTP computes the share of the master private key for the initial node by evaluating the master polynomial after which this is securely transmitted along with the set of witness values (step 1.1). Due to our incorporation of verifiable secret sharing, the initial node is capable of verifying whether the provided share is correct (steps 1.2 and 1.3). Every initial node that obtains a correct share of the master private key is capable of providing a partial key management service to nodes that wish to join the network during

Network Setup Algorithm

1. First, the TTP generates two large primes p and q such that $q|p-1$ and selects a generator g of cyclic subgroup $\mathbb{G} \subset \mathbb{Z}_p^*$ which has order q. The TTP also selects security parameter t, the threshold value that indicates the number of network nodes required to provide a successful key management service.
2. The TTP generates the master polynomial $f(x)$ of degree $t-1$ with randomly chosen coefficients $a_i \in \mathbb{Z}_q^*$:
$$f(x) = \sum_{i=0}^{t-1} a_i x^i \in \mathbb{Z}_q^*[x]. \quad (1)$$
3. The TTP defines the master private key $MSK = f(0)$, the master public key $MPK = g^{MSK} \pmod p$, the public witness values $\omega = \{w_i \equiv g^{a_i} \pmod p\}$ for $0 \leq i \leq t-1$.
4. The TTP defines three collision-free hash functions (h_1, h_2, h_3), a secure DLP-based signature SIG (e.g., Schnorr [31], DSA [32]) and encryption ENC (e.g., ElGamal [33]) scheme, a forge-resistant message authentication code MAC (e.g., HMAC [34, 35]), and a semantically secure symmetric encryption cipher CIP (e.g., AES [36]).
5. Finally, the TTP publishes the public network parameters (p, q, g, t, MPK, h_1, h_2, h_3, SIG, ENC, MAC, CIP) in a secure, public space.

Algorithm 1: The TTP executes the network setup algorithm to establish and define the network parameters.

1. Centralized Secret Share Establishment Protocol

1.1. First, the TTP computes the secret share ss_l for initial node N_l with identity ID_l as follows:
$$ss_l \equiv f(ID_l) \pmod q. \quad (2)$$
The TTP securely transmits secret share ss_l and witness values ω to node N_l.

1.2. Each initial node N_l computes the public share ps_l that should correspond to the received secret share ss_l as follows:
$$ps_l \equiv \prod_{i=0}^{t-1} {w_i}^{{ID_l}^i} \pmod p. \quad (3)$$

1.3. Finally, each initial node N_l verifies whether its obtained secret share ss_l is correct:
$$ps_l \stackrel{?}{\equiv} g^{ss_l} \pmod p. \quad (4)$$
If verification is successful, initial node N_l accepts the obtained secret share ss_l.

Protocol 1: The technical details of the centralized secret share establishment protocol.

the network operational phase. These partial key management services include the provisioning of (i) a partial secret share or (ii) a partial initial proxy key pair. A threshold amount of partial key management services converges to a successful key management service, as if this was directly provided by a centralized TTP. Our secret sharing construction is based on the work of Shamir [8] and the verifiability extension is based on the work of Feldman [11]. The technical details of the protocol is described in Protocol 1.

6.1.3. Centralized Proxy Key Pair Establishment

In this protocol, the TTP initiates an interactive protocol with each initial node to provide them with their initial proxy key pair and the associated initial commitment value. This initial commitment value consists of two partial commitments, one generated by the TTP (step 2.1) and one generated by the initial node (step 2.2). The initial commitment value is then bound by the TTP to the initial node's partial private proxy key (step 2.3). The initial node verifies whether the TTP provided an honest key management service by checking that the binding between its initial commitment and its partial private proxy key is correct (step 2.4). This binding technique in our protocol allows our scheme to benefit from a FD-TTP with trust level 3 (see section 7 for more details). Finally, the initial node computes its initial proxy key pair (step 2.5).

The initial public proxy key $PK_{l,0}$ and the initial private proxy key $SK_{l,0}$ are long-lasting keys and should never be disclosed. Instead, these are used as key derivation keys for issuing self-generated certificates. This is covered in more detail in Protocol 3, the certificate issuing and updating protocol. After the cluster of t initial nodes obtained their shares of the master private key and their initial keying material (i.e., the initial commitment, private proxy key and public proxy key), the network becomes self-organized and is no longer reliant on a centralized TTP. The TTP destroys the master polynomial and its secret coefficients after which it leaves the network. The technical details of our interactive centralized proxy key pair establishment protocol between the TTP and initial node N_l is described in Protocol 2.

6.2. Operational Phase

With a cluster of t nodes initialized by the TTP, the network enters the operational phase. During the operational phase,

2. Centralized Proxy Key Pair Establishment Protocol

2.1. First, the TTP selects a random secret value $sv_{TTP} \in \mathbb{Z}_q^*$ and computes partial commitment value c_{TTP} as follows:

$$c_{TTP} \equiv g^{sv_{TTP}} \pmod{p}. \tag{5}$$

The TTP securely transmits the partial commitment c_{TTP} to initial node N_l.

2.2. Initial node N_l selects a random secret value $sv_l \in \mathbb{Z}_q^*$ to obtain its own partial commitment c_l. Both partial commitments are then combined into its initial commitment $c_{l,0}$:

$$c_{l,0} \equiv c_l \cdot c_{TTP} \pmod{p} \equiv g^{sv_l} \cdot c_{TTP} \pmod{p}. \tag{6}$$

Initial node N_l securely transmits its initial commitment $c_{l,0}$ to the TTP.

2.3. The TTP computes the partial private proxy key SK_{TTP} as follows:

$$SK_{TTP} \equiv sv_{TTP} + MSK \cdot h_1(ID_l, c_{l,0}) \pmod{q}. \tag{7}$$

The TTP securely transmits partial private proxy key SK_{TTP} to initial node N_l.

2.4. Initial node N_l checks whether the TTP provided an honest key management service by verifying whether the TTP incorporated the same random secret value sv_{TTP} in the establishment of its partial commitment c_{TTP} as well as the partial private proxy key SK_{TTP}. Initial node N_l checks the following:

$$g^{SK_{TTP}} \stackrel{?}{\equiv} c_{TTP} \cdot MPK^{h_1(ID_l, c_{l,0})} \pmod{p}. \tag{8}$$

2.5. Finally, initial node N_l utilizes partial private proxy key SK_{TTP} in establishing its initial proxy key pair. Node N_l computes its initial private proxy key $SK_{l,0}$ and corresponding initial public proxy key $PK_{l,0}$ as follows:

$$SK_{l,0} \equiv sv_l + SK_{TTP} \pmod{q}, \tag{9}$$

$$PK_{l,0} \equiv g^{SK_{l,0}} \pmod{p}. \tag{10}$$

Protocol 2: The technical details of the centralized proxy key pair establishment protocol.

the network and the key management functions in a fully self-organized manner. We define a network node to be a fully initialized node that obtained its personalized share of the master private key and obtained its initial proxy key pair. By this definition, the network node is also member of the FD-TTP. The following protocols are executed during this phase:

1. *Certificate Issuing & Updating*: Network nodes can issue and periodically update their own public key certificate in a non-interactive fashion.
2. *Secure Channel Establishment*: Network nodes can exchange their self-generated certificates and verify the authenticity of received certificates. The network nodes establish secure communication channels using the public key on authenticated certificates.
3. *Distributed Proxy Key Pair Establishment*: Nodes can join the network by requesting a threshold amount of nearby network nodes for key management services. The nearby network nodes can collectively provide the joining node with its initial proxy key pair.
4. *Distributed Secret Share Establishment*: Nodes can join the network by requesting a threshold amount of nearby network nodes for key management services. The nearby network nodes can collectively provide the joining node with its secret share of the master private key, thereby joining the distributed TTP.
5. *Certificate Revocation*: Network nodes can update their self-generated certificate frequently, making certificate revocation redundant. This feature simplifies the key management design.

6.2.1. Certificate Issuing & Updating

In this protocol, network nodes issue or update their self-generated certificate in a non-interactive manner. To execute this protocol, a node must be in possession of its initial commitment and its initial proxy key pair. A network node first selects a random secret value from which it computes a certificate-specific commitment and a corresponding certificate-specific proxy key pair. This certificate-specific proxy key pair is essentially derived from its initial proxy key pair (step 3.1). For a network node N_l, its certificate will contain at least the following parameters: the identity of the network node, ID_l, its initial commitment $c_{l,0}$, the certificate-specific commitment c_l, the certificate-specific public proxy key PK_l and a timestamp TS. The network node uses these parameters and its certificate-specific private proxy key SK_l as inputs to generate a signature σ (step 3.2). The network node then adds the signature to define its self-generated certificate (step 3.3). The technical details of the certificate issuing and updating protocol is described in Protocol 3.

6.2.2. Secure Channel Establishment

This protocol describes the process of verifying exchanged self-generated certificates such that network nodes can estab-

3. Certificate Issuing & Updating Protocol

3.1. First, network node N_l selects a random secret value $sv_l \in \mathbb{Z}_q^*$. Then, the network node computes the corresponding commitment c_l, private proxy key SK_l and public proxy key PK_l as follows:

$$c_l \equiv g^{sv_l} \pmod{p}, \tag{11}$$

$$SK_l \equiv sv_l + SK_{l,0} \cdot h_1(ID_l, c_l) \pmod{q}, \tag{12}$$

$$PK_l \equiv g^{SK_l} \pmod{p}. \tag{13}$$

3.2. Network node N_l creates a signature σ on the certificate using its private proxy key SK_l. The contents of the certificate are the inputs to create the signature:

$$\sigma = SIG_{SK_l}(ID_l, c_{l,0}, c_l, PK_l, TS), \tag{14}$$

where timestamp TS represents the validity period or expiration time of the certificate.

3.3. Finally, network node N_l defines its self-generated certificate as:

$$CERT_l = \{ID_l, c_{l,0}, c_l, PK_l, TS, \sigma\}. \tag{15}$$

Self-generated certificate $CERT_l$ can be exchanged with other network nodes to establish secure unicast channels.

Protocol 3: The technical details of the certificate issuing & updating protocol.

4. Secure Channel Establishment Protocol

4.1. First, network nodes N_l and N_k exchange their self-generated certificates. The following three verification steps are executed by network node N_k to authenticate certificate $CERT_l$.

4.2. Network node N_k inspects timestamp TS to verify that certificate $CERT_l$ has not expired.

4.3. Network node N_k verifies whether the binding between identity ID_l, public proxy key PK_l, and commitments c_l and $c_{l,0}$ is correct by checking whether both terms in the equation underneath are equivalent:

$$PK_l \stackrel{?}{\equiv} c_l \cdot (c_{l,0} \cdot MPK^{h_1(ID_l, c_{l,0})})^{h_1(ID_l, c_l)} \pmod{p}. \tag{16}$$

4.4. Network node N_k verifies whether the information on the certificate has not been tampered with and that the certificate has been signed by the proxy private key that corresponds to public proxy key PK_l:

$$\sigma \stackrel{?}{=} SIG^{-1}{}_{PK_l}(ID_l, c_{l,0}, c_l, PK_l, TS), \tag{17}$$

where SIG^{-1} represents the verification algorithm of a signature scheme.

4.5. If network node N_k has been able to execute each verification step successfully, it computes two shared symmetric keys. The shared symmetric encryption and decryption key $K_{E(k,l)}$ and the shared symmetric signing and verification key $K_{S(k,l)}$:

$$K_{E(k,l)} = h_2(PK_l{}^{SK_k} \pmod{p}). \tag{18}$$

$$K_{S(k,l)} = h_3(PK_l{}^{SK_k} \pmod{p}). \tag{19}$$

These symmetric keys enable any two network nodes to securely communicate.

Protocol 4: The technical details of the secure channel establishment protocol.

lish a secure communication channel. Thus, we assume that a pair of network nodes exchanged their self-generated certificates with each other (step 4.1). The network node first inspects whether the received certificate is still valid by examining the timestamp (step 4.2). Then, the network node verifies whether the binding between the published identity, the public proxy key and the commitments is correct (step 4.3). If this verification step is successful, the network node can be confident that the public proxy key is created by the network node with the identity on the certificate. Finally, the network node verifies whether the information on the certificate has not been tampered with and that the signature is created with the private proxy key that corresponds to the public proxy key that is published on the certificate (step 4.4). If all these verification steps are successful, the network node can compute a pairwise symmetric key for encryption and decryption purposes, and a pairwise symmetric key for signing and verification purposes. The pairwise symmetric key is created from the network node's own private proxy key (that was used to sign the certificate that it transmitted to the other network node) and the public proxy key that was published on the received and verified certificate of the other network node (step 4.5). The technical details of the secure channel establishment protocol is described in Protocol 4.

This construction enables network nodes to perform authenticated encryption. The most secure form of authenticated encryption follows the Encrypt-then-MAC principle, in which a plaintext message is first encrypted after which the MAC is pro-

duced from the resulting ciphertext. The ciphertext and MAC are then sent together. This is the standard method according to ISO/IEC 19772:2009 [37] and is used in Internet Protocol Security (IPSec) [38].

6.2.3. Distributed Proxy Key Pair Establishment

Nodes are allowed to join the network during the operational phase. Assuming that at least a threshold amount of network nodes are within the transmission range of the joining node, key management services can be provided. In the distributed proxy key pair establishment protocol, a joining node starts by selecting a random temporary private key from which it computes the corresponding temporary public key (step 5.1). This temporary key pair allows the nearby servers to securely transmit data later on in the protocol. The nearby servers are informed of the joining node, prompting the establishment of the joining node's initial commitment value with partial commitments provided by the servers (step 5.2) as well as the joining node itself (step 5.3). The nearby servers are then informed of the joining node's initial commitment, from which the servers can compute partial private proxy keys (step 5.4). These partial private proxy keys are securely transmitted using the joining node's temporary public key. The joining node then verifies whether the partial private proxy keys have been honestly generated with the server's share of the master private key (step 5.5). Once the joining node received a threshold amount of correct partial private proxy keys, it can compute its initial private proxy key and its corresponding initial public proxy key (step 5.6). The technical details of the distributed proxy key pair establishment protocol, for joining node N_l and the set of nearby servers denoted by ϕ, are described in Protocol 5.

6.2.4. Distributed Secret Share Establishment

This protocol can directly follow the distributed proxy key pair establishment protocol such that the same set of servers can provide the joining node with its share of the master private key. In such case, the joining node previously received the self-generated certificates of these servers and have established secure unicast channels. For the servers to maintain the secrecy of their own share, they must perform a shuffling mechanism. Therefore, the joining node broadcasts the set of certificates from its nearby servers (step 6.1) of which every server chooses a random assisting server's certificate to perform the shuffling mechanism with (step 6.2). Each server generates a shuffle value that is securely transmitted to the assisting server (step 6.3). One server adds and the assisting server subtracts the shuffle value (step 6.4) and computes its shuffled partial secret share (step 6.5). These are securely transmitted to the joining node that combines them together (step 6.6) and verifies its correctness (step 6.7). The technical details of the distributed secret share establishment protocol, for joining node N_l and the set of nearby servers denoted by ϕ, are described in Protocol 6.

6.2.5. Certificate Revocation

Certificate revocation is an important key management feature and requires mechanisms for the following scenarios:

1. Network nodes require a mechanism in which they can revoke their own certificate when they believe that its key pair has been compromised.
2. Network nodes require a mechanism in which they can accuse and revoke the certificate of a network node which is behaving suspiciously and may have been compromised.
3. A mechanism is required which informs all the network nodes about recently revoked certificates.
4. A node which joins the network must also be provided with a list of revoked unexpired certificates.

It is important to notice that certificate revocation is important when certificates have become compromised long before their expiration date. Therefore, if we limit the period of time between certificate compromise and certificate expiration, certificate revocation becomes redundant [23, 39]. In our DISTANT scheme, we take advantage of the fact that self-generated certificates can be updated non-interactively. Therefore, we proposed that these certificates are updated frequently (e.g., daily) to limit the time between certificate compromise and certificate expiration. The frequent updating of self-generated certificates only causes a minor computational overhead increase while alleviating the key management from complicated and expensive certificate revocation mechanisms.

6.3. Share Updating Phase

The network alternates between the operational phase and the share updating phase. This phase only contains the share updating protocol in which every network node will be provided with a new and independent secret share of the master private key. This makes a collection of less than t secret shares, collected by a mobile adversary [30] in the previous operational phase, unusable in the reconstruction of the master private key. This phase is therefore necessary to maintain long-term security.

6.3.1. Share Updating

The share updating protocol is triggered by the mobile network or the software-defined controller. This entity can transmit a signal to the cluster head of a mobile small cell which then informs the members of that mobile small cell to initiate the share updating protocol. The protocol is initiated by having each network node within this cluster broadcast their self-generated certificate to establish secure unicast channels between the cluster nodes (step 7.1). Each cluster node then generates a random update polynomial with its leading coefficient being 0, allowing the update of secret shares without updating the master key pair (step 7.2). Based on the update polynomial, each cluster node computes update witness values (step 7.3) and partial update shares (step 7.4) for every other cluster node. These are securely transmitted and verified (step 7.5). Failing to verify the correctness of a partial update share prompts an accusation procedure. The details of this accusation procedure is covered in [13, 14]. When no accusation procedures are prompted, every cluster node computes its updated secret share (step 7.6) and the updated witness values (step 7.7). The technical details of the share updating protocol, initiated by

5. Distributed Proxy Key Pair Establishment Protocol

5.1. First, joining node N_l selects a random temporary private key $SK_l \in \mathbb{Z}_q^*$ and computes corresponding public key:

$$PK_l \equiv g^{SK_l} \pmod{p}. \tag{20}$$

Joining node N_l broadcasts its identity ID_l and temporary public key PK_l to nearby servers $N_k \in \phi$.

5.2. Server $N_k \in \phi$ selects a random secret value $sv_k \in \mathbb{Z}_q^*$ and computes the partial initial commitment value c_k:

$$c_k \equiv g^{sv_k} \pmod{p}. \tag{21}$$

Server $N_k \in \phi$ then broadcasts its certificate $CERT_k$, partial initial commitment c_k and witness values ω. The exchange of the joining node's temporary public key PK_l and the certificates $CERT_k$ of the nearby servers allows for the establishment of secure unicast channels between the joining node and its servers.

5.3. Joining node N_l selects a random secret value $sv_l \in \mathbb{Z}_q^*$ to compute its own partial commitment c_l. Its own partial commitment and t received partial commitments are then combined into its initial commitment $c_{l,0}$:

$$c_{l,0} \equiv g^{sv_l} \cdot \prod_{N_k \in \phi} c_k^{\lambda_k^\phi(0)} \pmod{p}, \tag{22}$$

where $\lambda_k^\phi(x)$ represents the Lagrange coefficient:

$$\lambda_k^\phi(x) \equiv \prod_{N_j \in \phi, k \neq j} \frac{x - ID_j}{ID_k - ID_j} \pmod{q}. \tag{23}$$

Joining node N_l broadcasts its initial commitment value $c_{l,0}$.

5.4. Server $N_k \in \phi$ computes the partial private proxy key SK_k as follows:

$$SK_k \equiv sv_k + ss_k \cdot h_1(ID_l, c_{l,0}) \pmod{q}, \tag{24}$$

Server $N_k \in \phi$ securely transmits partial private proxy key SK_k to joining node N_l.

5.5. Joining node N_l verifies whether each received partial private proxy key SK_k is correct by verifying the binding between corresponding partial private proxy keys SK_k and partial commitments c_k:

$$g^{SK_k} \stackrel{?}{\equiv} c_k \cdot ps_k^{h_1(ID_l, c_{l,0})} \pmod{p}, \tag{25}$$

where ps_k represents the public share of server N_k, described in Equation 3.

5.6. Finally, joining node N_l combines the t verified partial private proxy keys SK_K to establish its initial proxy key pair. Joining node N_l computes its initial private proxy key $SK_{l,0}$ and corresponding initial public proxy key $PK_{l,0}$:

$$SK_{l,0} \equiv sv_l + \sum_{N_k \in \phi} (SK_k \cdot \lambda_k^\phi(0)) \pmod{q}, \tag{26}$$

$$PK_{l,0} \equiv g^{SK_{l,0}} \pmod{p}. \tag{27}$$

The established proxy key pair does not require verification, since the partial private proxy keys were already verified.

Protocol 5: The technical details of the distributed proxy key pair establishment protocol.

a cluster of t network nodes denoted by ϕ, are described in Protocol 7.

After this cluster of t network nodes has their secret shares updated, they broadcast a notification such that other nearby network nodes can request to have their share updated. These shares are updated according to the distributed secret share establishment protocol. The requesting node is therefore required to be within transmission range of a cluster of t network nodes with updated secret shares. This process continues until every network node has their share updated.

7. Security Analysis

In this section, we prove that our key management scheme satisfies the proposed security objectives:

- our key management scheme is resilient against disruptive adversaries to establish a trustworthy key management service,
- our key management scheme is resilient against mobile adversaries to provide long-term network security, and
- our key management scheme reaches trust level 3 to provide an additional layer of security against network compromise.

Furthermore, we present the simulation results that give insight into selecting a proper security threshold for various network densities. This is important since every urban environment is unique and has their own individual mobile node densities. Therefore, a proper security threshold must be chosen such that the security level of the network is maximized while also guaranteeing that every network node has enough servers within its transmission range to be provided with key management services.

6. Distributed Secret Share Establishment Protocol

6.1. First, joining node N_l broadcasts the set of certificates from its nearby servers $N_k \in \phi$:

$$CERT_\phi = \{CERT_k \mid N_k \in \phi\}. \tag{28}$$

6.2. Server $N_k \in \phi$ selects a random server $N_j \in \phi$ to perform the shuffling mechanism with. Server N_k selects a random shuffle value $\delta_{k,j} \in \mathbb{Z}_q^*$ and defines the random shuffle value for N_j as:

$$\delta_{j,k} = -\delta_{k,j} \pmod q. \tag{29}$$

Server N_k encrypts $\delta_{j,k}$ such that only server N_j can decrypt it. Server N_k then securely transmits a 3-tuple with its identity ID_k, the identity of the assisting server ID_j and the encrypted shuffle value $ENC(\delta_{j,k})$ to joining node N_l.

6.3. Joining node N_l broadcasts the set of encrypted shuffle values:

$$\delta_\phi = \{(ID_k, ID_j, ENC(\delta_{j,k})) \mid N_k \in \phi\}. \tag{30}$$

6.4. Server $N_k \in \phi$ inspects the received tuples and decrypts any encrypted shuffle values that are intended for it. Server $N_k \in \phi$ then sums up all collected shuffle values:

$$\delta_k \equiv \delta_{k,j} + \sum_{N_i \in \phi} \delta_{k,i} \pmod q. \tag{31}$$

6.5. Server $N_k \in \phi$ computes the shuffled partial secret share $ss_{k,l}$:

$$ss_{k,l} \equiv ss_k \cdot \lambda_k^\phi(ID_l) + \delta_k \pmod q, \tag{32}$$

where $\lambda_k^\phi(x)$ represents the Lagrange coefficient as defined in Equation 23. Server N_k then securely transmits the shuffled partial secret share $ss_{k,l}$ and witness values ω to joining node N_l.

6.6. Joining node N_l combined the t shuffled partial secret shares to obtain its secret share ss_l:

$$ss_l \equiv \sum_{N_k \in \phi} ss_{k,l} \pmod q. \tag{33}$$

6.7. Finally, joining node N_l verifies whether its secret share ss_l is correct:

$$g^{ss_l} \equiv ps_l \pmod p \equiv \prod_{i=0}^{t-1} {w_i}^{{ID_l}^i} \pmod p, \tag{34}$$

where ps_l represents the public share of N_l and w_i represent the public witness values.

Protocol 6: The technical details of the distributed secret share establishment protocol.

7.1. Security Evaluation against Disruptive Adversaries

In the following three theorems, we prove that our key management scheme is resilient against disruptive adversaries providing false key management services. This applies to both the key management service in which nodes request their share of the master private key and the key management service in which nodes request the establishment of their initial proxy key pair.

Theorem 1. *Any individual network node N_l can verify that the provided key management service from the distributed secret share establishment protocol by the set of servers $N_k \in \phi$ where $|\phi| \geq t$ has been trustworthy, i.e., the network node N_l can verify that obtained secret share ss_l is correct.*

Proof. We prove this theorem by showing that network node N_l can utilize the public witness values $w_i \in \omega$ to pre-compute its public share ps_l. The key management service has been trustworthy if and only if the provided secret share ss_l corresponds to the pre-computed public share ps_l. This proves that the servers $N_k \in \phi$ used their secret share and correctly applied the shuffling values during the execution of the distributed secret share establishment protocol. We prove that the network node N_l is able to verify this mathematical correspondence between its public share ps_l, secret share ss_l and the public witness values $w_i \in \omega$ through a series of mathematical equivalences:

$$ps_l \equiv \prod_{i=0}^{t-1} {w_i}^{{ID_l}^i} \pmod p \tag{42}$$

$$\equiv \prod_{i=0}^{t-1} g^{a_i \cdot {ID_l}^i} \pmod p \tag{43}$$

$$\equiv g^{\sum_{i=0}^{t-1} a_i \cdot {ID_l}^i} \pmod p \tag{44}$$

$$\equiv g^{\sum_{N_k \in \phi} (ss_k \cdot \lambda_k^\phi(ID_l))} \pmod p \tag{45}$$

$$\equiv g^{\sum_{N_k \in \phi} (ss_k \cdot \lambda_k^\phi(ID_l) + \delta_k)} \pmod p \tag{46}$$

$$\equiv g^{ss_l} \pmod p \tag{47}$$

□

Theorem 2. *Any individual network node N_l can verify that the provided key management service from the distributed proxy key pair establishment protocol by the set of servers $N_k \in \phi$ where $|\phi| \geq t$ has been trustworthy, i.e., the network node N_l can verify that obtained initial proxy key pair $(PK_{l,0}, SK_{l,0})$ is correct.*

Proof. We prove this theorem in a similar fashion as we did for Theorem 1. We prove that the key management service has

7. Share Updating Protocol

7.1. First, each network node $N_l \in \phi$ broadcasts its certificate $CERT_l$ such that every network node within the cluster can establish secure unicast channels with each other.

7.2. Each network node $N_l \in \phi$ generates an update polynomial $f_l(x)$ of degree $t-1$ with random coefficients a_i from finite field $\mathbb{Z}_q^*$ and leading coefficient $a_0 = 0$:

$$f_l(x) = \sum_{i=1}^{t-1} a_i x^i \in \mathbb{Z}_q^*[x]. \tag{35}$$

7.3. Each network node $N_l \in \phi$ computes the $t-1$ corresponding update witness values $w_{i,l}$. The set of update witness values generated by network node N_l is denoted as ω_l:

$$\omega_l = \{w_{i,l} \equiv g^{a_i} \pmod p\} \text{ for } 1 \leq i \leq t-1. \tag{36}$$

7.4. Each network node $N_l \in \phi$ computes partial update shares $ss_{l,k}$ for every network node $N_k \in \phi$ (including itself) and encrypts these such that only intended network node N_k can retrieve it:

$$ss_{l,k} \equiv f_l(ID_k) \pmod q. \tag{37}$$

$$e_{l,k} \equiv CIP_{K_{E(l,k)}}(ss_{l,k}) \text{ for all } k \neq l. \tag{38}$$

Then, network node N_l broadcasts a message containing its identity ID_l, the set of update witness values ω_l, the set of encrypted partial update shares $e_{l,k}$ and a signature.

7.5. Each network node $N_l \in \phi$ decrypts the update shares intended for N_l and verifies whether the obtained partial update shares $ss_{k,l}$ are correct:

$$g^{ss_{k,l}} \stackrel{?}{\equiv} \prod_{i=1}^{t-1} {w_{i,k}}^{{ID_l}^i} \pmod p. \tag{39}$$

7.6. Each network node $N_l \in \phi$ computes its updated secret share by combining its current secret share with the t partial update shares.

$$ss_l \equiv ss_l + \sum_{N_k \in \phi} ss_{k,l} \pmod q. \tag{40}$$

7.7. Each network node $N_l \in \phi$ computes the updated witness values by combining the current witness values with the t sets of update witness values and redefines ω as:

$$\omega = \{w_0, w_i \equiv w_i \cdot \prod_{N_k \in \phi} w_{i,k} \pmod p\} \text{ for } 1 \leq i \leq t-1. \tag{41}$$

Protocol 7: The technical details of the share updating protocol.

been trustworthy if and only if every server $N_k \in \phi$ used their secret share ss_k in the establishment of the partial private keys SK_k. We prove that the network node N_l is able to verify this mathematical correspondence between its intial public proxy key $PK_{l,0}$, its initial private proxy key $SK_{l,0}$ and its initial commitment $c_{l,0}$ through a series of mathematical equivalences:

$$\begin{aligned} PK_{l,0} &\equiv c_{l,0} \cdot MPK^{h_1(ID_l, c_{l,0})} \pmod p, && (48)\\ &\equiv g^{sv_l} \cdot g^{\sum_{N_k \in \phi}(sv_k \cdot \lambda_k^{\phi}(0))} \cdot g^{MSK \cdot h_1(ID_l, c_{l,0})} && (49)\\ &\equiv g^{sv_l} \cdot g^{\sum_{N_k \in \phi}((sv_k + ss_k \cdot h_1(ID_l, c_{l,0})) \cdot \lambda_k^{\phi}(0))} && (50)\\ &\equiv g^{sv_l + \sum_{N_k \in \phi}(SK_k \cdot \lambda_k^{\phi}(0))} \pmod p && (51)\\ &\equiv g^{SK_{l,0}} \pmod p && (52) \end{aligned}$$

□

Theorem 3. *Any individual network node N_l can detect which server(s) $N_k \in \phi$ where $|\phi| \geq t$ provided a malicious key management service during the execution of the distributed proxy key pair establishment protocol.*

Proof. We prove this theorem in a similar fashion as we did for Theorem 1 and Theorem 2. First, network node N_l knows the partial commitment c_k and is able to compute the public share ps_k of every server $N_k \in \phi$. This allows network node N_l to compute the partial public proxy key PK_k that corresponds to the partial private proxy key SK_k that server $N_k \in \phi$ is supposed to provide. We prove that the key management service from server $N_k \in \phi$ has been trustworthy if and only if server $N_k \in \phi$ honestly incorporated their secret share ss_k in computing the partial private key SK_k through the following series of mathematical equivalences:

$$\begin{aligned} PK_k &\equiv c_k \cdot {ps_k}^{h_1(ID_l, c_{l,0})} \pmod p && (53)\\ &\equiv g^{sv_k} \cdot (g^{ss_k})^{h_1(ID_l, c_{l,0})} \pmod p && (54)\\ &\equiv g^{sk_k + ss_k \cdot h_1(ID_l, c_{l,0})} \pmod p && (55)\\ &\equiv g^{SK_k} \pmod p && (56) \end{aligned}$$

□

7.2. Security Evaluation against Mobile Adversaries

In the following two theorems, we prove that our key management scheme is resilient against mobile adversaries under the assumption that share updating phases are executed prior to any mobile adversary compromising and extracting a threshold number of secret shares. We prove that the updated secret shares are a correct sharing of the master private key MSK and that a mobile adversary that collects fewer than t secret shares in between share updating phases is unable to reconstruct the master private key MSK.

Theorem 4. *The updated secret shares, created at the end of each share updating phase, can be used to reconstruct the original master private key MSK.*

Proof. This is a two-part proof. First, we show that the updated secret share of any arbitrary network node N_l, denoted by ss_l^{new}, is defined through the evaluation of a random polynomial with the master private key MSK as its leading coefficient. We prove this through the following series of mathematical equivalences:

$$ss_l^{new} \equiv ss_l + \sum_{N_k \in \phi} ss_{k,l} \pmod q \tag{57}$$

$$\equiv f(ID_l) + \sum_{N_k \in \phi} f_k(ID_l) \pmod q \tag{58}$$

$$\equiv MSK + \sum_{i=1}^{t-1} (a_i + \sum_{N_k \in \phi} a_i^{new}) \cdot x^i \tag{59}$$

$$\equiv MSK + \sum_{i=1}^{t-1} b_i \cdot x^i \pmod q \tag{60}$$

As per usual, the master private key MSK can be reconstructed by combining the secret shares ss_k from at least a threshold amount of servers $N_k \in \phi$ through Lagrange interpolation:

$$MSK \equiv \sum_{N_k \in \phi} ss_k \cdot \lambda_k^{\phi}(0) \pmod q, \tag{61}$$

where $\lambda_k^{\phi}(x)$ represents the Lagrange coefficient:

$$\lambda_k^{\phi}(x) \equiv \prod_{N_j \in \phi, k \neq j} \frac{x - ID_j}{ID_k - ID_j} \pmod q. \tag{62}$$

□

Theorem 5. *An adversary who knows less than the threshold number of secret shares before any share updating period, cannot determine the master private key MSK.*

Proof. We prove this theorem by contradiction. Assuming that the adversary gathered $m < t$ secret shares in between two consecutive share updating periods. From the set of m secret shares $\{ss_1, \cdots, ss_m\}$, we construct the following system of linear equations:

$$\begin{cases} ss_1 = MSK + a_1 \cdot ID_1 + a_2 \cdots + a_{t-1} \cdot {ID_1}^{t-1} \\ ss_2 = MSK + a_1 \cdot ID_2 + a_2 \cdots + a_{t-1} \cdot {ID_2}^{t-1} \\ \quad \vdots \\ ss_m = MSK + a_1 \cdot ID_m + a_2 \cdots + a_{t-1} \cdot {ID_m}^{t-1} \end{cases}$$

Based on the fundamental theorem of linear algebra, it is not possible to solve a system of m linearly independent equations with t unknowns where $m < t$. Therefore, the adversary is unable to determine the master private key MSK with fewer than t secret shares. □

7.3. Security Evaluation against a compromised distributed TTP

In the following two theorems, we prove that a compromised and malicious TTP is unable to compute the any node's private key and is unable to launch identity impersonation attacks without being detected. This means that the FD-TTP in our key management scheme reaches trust level 3.

Theorem 6. *A compromised and malicious FD-TTP is unable to (easily) compute the initial private proxy key $SK_{l,0}$ of any network node N_l.*

Proof. This theorem can be proven through the inspection of the construction of the initial private proxy key $SK_{l,0}$ of any network node N_l:

$$SK_{l,0} \equiv sv_l + sv_{TTP} + MSK \cdot h_1(ID_l, c_{l,0}) \pmod q \tag{63}$$

The secret value sv_l of network node N_l is never disclosed to the malicious TTP. The malicious TTP is only able to estimate the secret value sv_l by solving the equation underneath. However, this is equivalent to solving the discrete logarithm problem. Therefore, we conclude that the malicious TTP is unable to (easily) compute a node's private proxy key.

$$c_{l,0} \equiv c_l \cdot c_{TTP} \pmod p \equiv g^{sv_l + sv_{TTP}} \pmod p \tag{64}$$

□

Theorem 7. *A compromised and malicious FD-TTP is unable to launch identity impersonation attacks without being detected.*

Proof. As stated by Al-Riyami et al. [26], a scheme that is based on certificateless PKC can either reach trust level 2 or trust level 3 depending on the key generation technique. The key generation technique that we employ includes a binding between the initial private proxy key $SK_{l,0}$ and the initial commitment $c_{l,0}$ for the arbitrary network node N_l. This binding effectively restricts network nodes as they can only create a single correct initial proxy key pair. This unique proxy key pair is then used to issue self-generated certificates that are linked to that initial proxy key pair.

For the malicious TTP, it is statistically impossible to generate the same unique initial proxy key pair as generated by network node N_l. The creation of an alternative initial private proxy key $SK_{l,0}^*$ that is linked to an alternative initial commitment $c_{l,0}^*$ would lead to an alternative set of self-generated certificates. Identity impersonation attacks from a malicious TTP can be detected since there would be self-generated certificates circulating for the node N_l with identity ID_l, but one of the self-generated certificates would contain the original commitment $c_{l,0}$ and the other would contain the *fake* commitment $c_{l,0}^*$. This

indicates malicious activity from the TTP. Therefore, in our key management design, a compromised and malicious TTP is unable to launch identity impersonation attacks without being detected. □

7.4. *Security Threshold versus Key Management Service Availability*

The deployment of mobile small cells in urban environment requires appropriate network parameters, such as the security threshold or the time period in between share updating phases. Every urban environment is unique with their particular mobile node densities and distributions. Therefore, we ran numerous simulations for eight urban environments (eight of the largest Portuguese cities) to investigate how the key management service availability is affected by the security threshold. For the execution of these simulations, we made the following assumptions:

- For each urban environment, we computed the population density and made the assumption that every citizen possesses, on average, one mobile device. Therefore, we utilized a one-to-one correspondence between the number of citizens and the number of mobile devices.
- To estimate the number of servers that are within the transmission range of a mobile device, we assumed that the transmission range of every mobile device is equal. Furthermore, we based the transmission range on the predicted size of the mobile small cell. As stated in section 3.1, a mobile small cell has an approximate radius of 50 to 100 meters. If we assume that the hotspot can be at the edge of its mobile small cell but is still able to control its cluster through D2D communication, it requires a range of approximately 100 to 200 meters to reach all its cluster members. We selected the transmission range of every mobile device to be the average of that, thus 150 meters.
- The simulated mobile devices were randomly distributed throughout the network.

We ran simulations for eight of the largest cities in Portugal (Lisbon, Vila Nova de Gaia, Porto, Braga, Amadora, Almada, Coimbra and Funchal) with a population density[2] varying between 7,365 to 449 people per square kilometer. We assumed that every citizen has, on average, one mobile device and thus simulated networks with node densities that vary between 7,365 to 449 mobile devices per square kilometer. Networks that have a higher node density leads to network nodes having more servers within their transmission range to provide key management services. Therefore, the security threshold can be set higher in more dense networks without affecting the key management service availability of its network nodes. These findings are graphically represented in Figure 3.

In order to provide sufficient network nodes with an available key management service, we recommend that about 95% to 98% of all network nodes should have enough servers within its transmission range due to the steep reduction in key management service availability for a slight increase in security. We collected the relevant data for each of the eight urban environments and summarized them in Table 2, sorted by network density. Notice that the network density and the recommended security thresholds to provide 98%, 95%, 90% and 80% key management service availability seem to have a linear relationship. For any particular node density, the recommended security threshold that guarantees approximately 98% key management service availability is about $1/20^{\text{th}}$. This linear relationship also means that with a doubling of the network density, the recommended security threshold is about twice as large.

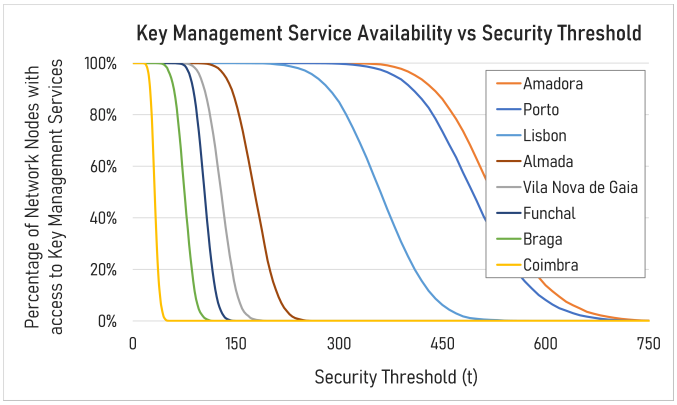

Figure 3: The estimated key management service availability for varying security threshold levels for eight of the largest Portuguese cities.

Table 2: Simulation results for the deployment of the mobile small cell networking scenario in eight Portuguese cities. The security thresholds maximize the security level while simultaneously guaranteeing a high chance of obtaining a successful key management service.

City	Density (pop./km^2)	Threshold t to reach KMS avail. 80%	90%	95%	98%
Amadora	7,365	466	436	412	388
Porto	6,943	436	406	381	352
Lisbon	5,053	310	285	264	241
Almada	2,479	154	144	134	124
Vila Nova de Gaia	1,794	112	104	98	91
Funchal	1,469	93	88	83	78
Braga	1,050	65	60	55	50
Coimbra	449	28	25	23	21

To provide proper security thresholds for other urban environments, we extrapolate our data. This enables the network initiator of any urban environment with its particular node density to select proper security thresholds that provide the highest level of security while simultaneously guaranteeing that a large percentage of the network nodes have access to obtaining their required key management services. These results are graphically presented in Figure 4.

Keep in mind that the security threshold also affects the overhead, thus it may not always be necessary to maximize the

[2]The population densities are computed from the estimated population and city area according to Wikipedia.

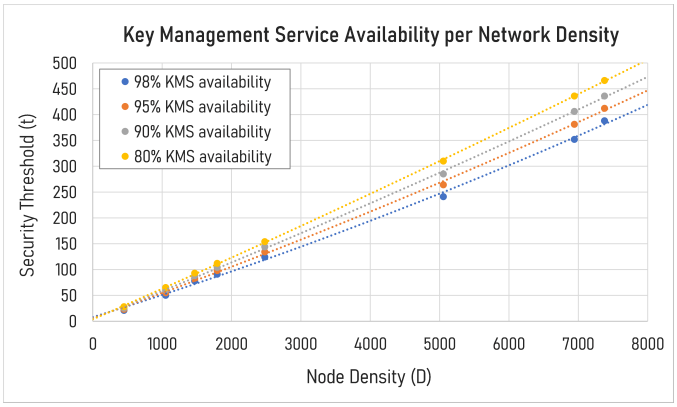

Figure 4: The relationship between network densities and the security threshold that guarantees a high percentage of network nodes to have access to key management services.

security threshold. For example, a high security threshold increases the communication overhead since more servers are requested to respond to key management service requests. Similarly, the computational overhead increases since more partial keying materials must be verified and combined in the establishment of the initial proxy key pair or the secret shares. Instead, the densest urban environments may prefer to reduce the security threshold and instead decide to have the secret shares updated more frequently. For example, if we assume that a mobile adversary is capable of compromising a stable number of mobile devices per time period, then the network initiator may prefer to half the security threshold in exchange for share updating phases that occur twice as frequent.

8. Performance Evaluation

8.1. Performance Comparison

This section will analyze and compare the communication overhead for the various protocols in FD-TTP-based key management schemes. We define the communication overhead as the number of unique messages which are transmitted to complete a protocol and is independent of the number of receivers. With this definition, we consider a unicast, a multicast and a broadcast message to contribute to the communication overhead by one unique message as these are single transmissions (even though the number of receivers vary). The estimated communication overheads can depend on the total number of network nodes n, the selected security threshold t, the average distance between two arbitrary network nodes d and the number of hops m to flood a local area with messages. The communication overheads per protocol per scheme is summarized in Table 3. Unfortunately, it is not possible to compare these protocols solely by their communication overhead, as some of them are insecure and require additional overhead in order to make them secure.

8.1.1. Initial Share & Key Establishment Protocols

For the initial share establishment protocol and the initial key establishment protocol, we assume that a cluster of t nodes are initialized. We estimate the communication overhead to be the number of transmissions required to provide every initial node with their share of the master private key and their initial key pair.

In our protocol design, we assume that a centralized TTP initializes the network. This assumption was also made in [15, 16, 20, 22]. In each protocol, the centralized TTP would generate a random master polynomial, define the master key pair, compute the secret share of every initial node and transmit these by unicast. Therefore resulting in an overhead of the order of t. The existence of a centralized TTP to initialize the network was not assumed in [18, 19, 21, 23]. In their protocols, the master key pair and secret shares are established in a distributed fashion. These protocols have a communication overhead of the order of t^2.

These protocols are executed only once (during network initialization) and therefore barely contribute to the communication overhead of the key management scheme as a whole. Furthermore, it is unfair to compare the efficiency of these protocols as their overhead mainly depends on the assumption whether a centralized TTP could perform the network initialization.

8.1.2. Key Updating Protocol

For the key updating protocol, we assume that the network contains n network nodes. We estimate the communication overhead to be the number of transmissions required for every network node to update their key.

We proposed a non-interactive key updating protocol, as well as [22, 23], allowing a communication overhead of 0. The other key management schemes require interaction between a network node and a threshold amount of servers. The key updating protocol from Luo et al. [15, 16] has an overhead of $n \times t$ and the key updating protocol from da Silva et al. [19] has an overhead of $n + t$.

8.1.3. Secure Channel Establishment Protocol

For the secure channel establishment protocol, we denote the average number of hops between two arbitrary network nodes as d and is dependent on the network size. We estimate the communication overhead to be the number of transmissions required for two arbitrary nodes to establish a secure channel.

Our protocol design, as well as the protocol designs from [15, 16, 20, 21, 22, 23], two network nodes are required to exchange their keying information to establish a secure channel. The keying material from both nodes therefore traverses on average d hops, leading to a communication overhead of $2 \times d$. The secure channel establishment protocols of [18, 19] have a communication overhead of 0, since their key management schemes are based on identity-based PKC.

8.1.4. Distributed Key Establishment Protocol

For the distributed key establishment protocol, we assume that exactly t servers are within transmission range of a joining node. With this assumption, we do not have to bother which t servers out of many are contributing to the establishment of the

Table 3: The communication overhead comparison table of DISTANT and related FD-TTP-based key management schemes. The communication overhead is, described in the number of message exchanges necessary to execute the entire protocol.

FD-TTP-based Key Management Scheme	Initial Share Establishm. Protocol	Initial Key Establishm. Protocol	Key Updating Protocol	Secure Channel Establishm. Protocol	Distributed Key Establishm. Protocol	Distributed Share Establishm. Protocol	Key Revocation Protocol	Share Updating Protocol
Luo et al. [15, 16]	$t+1$	t	nt	$2d$	$t+1$	$2t+2$	$(m-1)^2t^2+t$	$3nt+4n+t$
Deng et al. [18]	t^2	t^2	-	0	$n+t+1$	$t+1$	-	-
da Silva et al. [19]	t^2	t^2	$n+t$	0	$t+1$	$t+1$	$nt+n+t^2-2t-1$	-
Zhang et al. [20]	t	t	-	$2d$	$t+1$	$t+1$	-	-
Li et al. [21]	t^2	t^2	-	$2d$	$t+1$	$t+1$	-	$dnt+dn-dt$
Gharib et al. [22]	$t+1$	t	0	$2d$	$5t+1$	$t+1$	0	-
Lai et al. [23]	t^2	$2t^2$	0	$2d$	$2t+2$	$t+1$	0	-
DISTANT	t	$3t$	0	$2d$	$t+3$	$2t+2$	0	$2nt+3n-t^2-2t$

joining node's key. We estimate the communication overhead to be the number of transmissions required to provide a joining node with its initial key pair.

In our protocol design, a joining node initiates the interactive protocol by broadcasting a request to the t nearby servers. These t servers then broadcast a reply with partial commitment values. The joining node combines these with its own and then broadcasts its initial commitment value. The t nearby servers compute partial private keys and securely transmit these to the joining node. Thus, our protocol has a communication overhead of $t+3$.

The distributed key establishment protocol of the other key management schemes follow a similar progression, in which public information can be broadcasted and t secure unicast transmissions from the servers are required. The protocols from [15, 16, 19, 20, 21] benefit from a lower communication overhead of $t+1$, since the distributed TTP is directly able to provide partial keys. On the other hand, the protocols from Gharib et al. [22] and Lai et al. [23] have multiple rounds of secure exchanges, leading to a communication overhead of $5t+1$ and $2t+2$, respectively. Interestingly, Deng et al. [18] mentioned that a joining node has to disseminate its public key to every network node, which seems ominous as their scheme is based on identity-based PKC. This causes their protocol to have a communication overhead of $n+t+1$.

8.1.5. Distributed Share Establishment Protocol

For the distributed share establishment protocol, we assume that exactly t servers are within transmission range of a joining node. With this assumption, we do not have to bother which t servers out of many are contributing to the establishment of the joining node's share. We estimate the communication overhead to be the number of transmissions required to provide a joining node with its share.

In our protocol design, as well as [15, 16], a joining node initiates the interactive protocol by broadcasting a request to the t servers within transmission range. These t servers then reply with encrypted and signed shuffle values. The joining node combines these in a single message and broadcasts these shuffle values. The t network nodes decrypt and verify the shuffle values intended for them and incorporate these in their computation of a shuffled partial share. Each of the t servers now securely transmits these to the joining node. Thus, our protocol has a communication overhead of $2t+2$.

The protocols as proposed in [18, 20, 21, 22, 23] are incomplete and insecure. They merely proposed that a joining node would broadcast a request to which t servers would respond with partial shares, leading to a communication overhead of only $t+1$. They did mention that some form of shuffling must be incorporated to protect the secrecy of their shares, but did not provide any details how to do this. The protocol as proposed by da Silva et al. [19] is similar to the previously mentioned, however their protocol does not require a shuffling mechanism as its master polynomial is a bivariate polynomial [40]. Thus, their protocol has a reduced communication overhead of $t+1$ while also secure.

8.1.6. Key Revocation Protocol

For the key revocation protocol, we assume that each network node has t servers within its transmissions range, that at least t accusations are required to convict a network node for being malicious, and we denote m to be the number of hops in which a local flooding of accusation or revocation messages is taking place. We estimate the communication overhead to be the number of transmissions required to accuse, convict and disseminate the conviction of a network node.

In our protocol design, as well as [22, 23], network nodes are able to update their keying information periodically and in a non-interactive manner. With frequent updates, there is only a short time period between compromise and expiration. This makes certificate revocation redundant. Thus, our protocol and the protocols in [22, 23] have a communication overhead of 0.

Luo et al. [15, 16] proposed that a network node is effectively removed from the network once its keying information expires. Their key revocation protocol aims at preventing a malicious node from updating its key. When a network node detects malicious behavior, it sends an accusation message to every network node within its m-hop neighborhood. The value of m depends

on the time before the malicious node's key expires. Once network nodes receive at least a threshold amount of accusations, they consider the accused node malicious and reject any of its key updating requests. The communication overhead of their key revocation protocol is estimated at $(m-1)^2t^2 + t$. The key revocation protocol by da Silva et al. [19] also makes use of accusations and convictions. Any accusation would be broadcasted to every honest node by means of flooding and broadcast encryption. Once a network node is accused at least some threshold amount of times, a coalition of t servers creates and signs a conviction message which is then broadcasted through the network. The communication overhead of their key revocation protocol is estimated at $nt + n + t^2 - 2t - 1$.

8.1.7. Share Updating Protocol

For the share updating protocol, we denote the average number of hops between two arbitrary network nodes as d and is dependent on the network size. We estimate the communication overhead to be the number of transmissions required for every network node to update their share of the master private key.

In our protocol design, a cluster of t servers broadcast their self-generated certificates which allow them to securely transmit partial update shares. Each server creates an update polynomial, computes partial update shares and update witnesses, and securely transmits these. Finally, each server which has their share updated broadcasts a notification that nearby nodes can request to have their share updated. The nearby nodes essentially follow the distributed share establishment protocol to obtain their updated share. This generates a communication overhead of $2nt + 3n - t^2 - 2t$.

Luo et al. [15, 16] also proposed a scalable share updating protocol in which a cluster of t network nodes collaboratively create an encrypted and signed update polynomial. This update polynomial is then propagated through the network. Each network node can then send a local share updating request in which t servers collaboratively provide the network node with its update share. This protocol has an estimated communication overhead of $3nt + 4n + t$. Li et al. [21] proposed a less scalable share updating protocol in which t random nodes are selected to provide every other network node with its partial update shares. We estimated the communication overhead of this protocol to be $dnt + dn - dt$.

Based on our simulation data from section 7.4, we can approximate the communication overhead of our protocol as $(2/2000)n^2$, the protocol from Luo et al. [15, 16] as $(3/2000)n^2$ and the protocol from Li et al. [21] as $(d/2000)n^2$. Based on our assumed transmission range of 150 meters, the protocol from Li et al. [21] would be considered the most efficient for a network size that is smaller than 600m^2 and the most inefficient for a network size that is larger than 900m^2. For our purpose, we can conclude that our share updating protocol is the most efficient.

8.2. Discussion

In our key management scheme, we divide the master private key MSK into shares using a univariate polynomial. This has the disadvantage that the distributed share establishment protocol requires a shuffling mechanism to protect the secrecy of the servers' secret shares [15, 18, 21, 22, 23]. This shuffling mechanism requires additional interaction and computation which may be avoided with the use of bivariate polynomials [19, 40, 41]. However, the adaptation from univariate to bivariate polynomials is not trivial as we still require verifiability and proactivity. An additional benefit is that each partial secret share can be verified for correctness, therefore directly able to prove the malicious behavior from a server.

As mentioned previously, our key management scheme is designed to have a low overhead with the emphasis on communication overhead. As technology keeps improving, we did not consider computational overhead and memory storage overhead to be a significant restraint for mobile small cell networks. However, the computational overhead and memory storage overhead may be reduced by redesigning the protocols such that security is based on the discrete logarithm problem in the elliptic curve group [22]. We plan to examine the computational overhead of our DISTANT scheme in a future work and whether we can improve our design from a computational overheads perspective based on the mentioned protocol redesign strategy.

Finally, we proposed this key management scheme to secure multi-hop wireless D2D communications between nodes within a network of mobile small cells and a high node density. Based on this scenario, we assume that every network node has a connection with the cellular network. However, the 3GPP has also proposed the use-case in which network nodes are on the edge of cellular coverage or entirely outside of coverage [42]. If these nodes are unable to rely on any network infrastructure, this network of nodes essentially has a MANET structure. Since we assume existence of network infrastructure and rely on this network infrastructure to provide routing information to connect communicating nodes, we did not incorporate any routing mechanisms. Zhao et al. [43] proposed KM-SR, a key management and secure routing integrated framework for MANETs with a distributed TTP and would be an excellent resource to extend DISTANT to incorporate its own routing mechanism.

9. Conclusions

A mobile networking scenario which incorporates the use of mobile small cells can provide major advantages in delivering a high quality of service. However, the introduction of mobile small cells raise various security challenges. Cryptographic security solutions are capable of solving these as long as they are supported by an appropriate key management scheme. This article proposes DISTANT, the first secure key management scheme which is particularly designed for this mobile networking scenario to effectively and efficient support cryptographic security solutions. Our key management scheme relies on threshold secret sharing to decentralize trust and utilizes the self-generated certificates paradigm as a means to provide a high level of security while keeping the overheads to a minimum. The key management scheme has been evaluated and compared with seven related key management schemes from

both a security and communication overhead perspective. We found that our key management scheme reaches the highest level of security against the appropriately considered adversarial model. Furthermore, the design of the protocols are shown to be scalable and enables our key management scheme to efficiently support a network which covers dense urban environments.

Acknowledgements

The research work leading to this publication has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement H2020-MSCA-ITN-2016-SECRET-722424.

References

[1] Cisco, Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2012-2017, Tech. rep., Cisco, San Jose, CA, USA (2013).

[2] Cisco, Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017-2022, Tech. rep., Cisco, San Jose, CA, USA (2019).

[3] T. Q. S. Quek, G. de la Roche, I. Güvenç, M. Kountouris, Small Cell Networks: Deployment, PHY Techniques, and Resource Management, 1st Edition, Cambridge University Press, Cambridge, UK, 2013. doi:10.1017/CBO9781139061421.

[4] J. Rodriguez, A. Radwan, C. Barbosa, F. H. P. Fitzek, R. A. Abd-Alhameed, J. M. Noras, S. M. R. Jones, I. Politis, P. Galiotos, G. Schulte, A. Rayit, M. Sousa, R. Alheiro, X. Gelabert, G. P. Koudouridis, SECRET - Secure Network Coding for Reduced Energy Next Generation Mobile Small Cells: A European Training Network in Wireless Communications and Networking for 5G, in: Proceedings of the 7th International Conference on Internet Technologies and Applications (ITA), IEEE, Wrexham, UK, 2017, pp. 329–333. doi:10.1109/ITECHA.2017.8101964.

[5] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, J. Crowcroft, XORS in the Air: Practical Wireless Network Coding, IEEE/ACM Transactions on Networking 16 (3) (2008) 497–510. doi:10.1109/TNET.2008.923722.

[6] M. de Ree, G. Mantas, A. Radwan, S. Mumtaz, J. Rodriguez, I. E. Otung, Key Management for Beyond 5G Mobile Small Cells: A Survey, IEEE Access 7 (2019) 59200–59236. doi:10.1109/ACCESS.2019.2914359.

[7] M. de Ree, G. Mantas, J. Rodriguez, I. E. Otung, Distributed Trusted Authority-based Key Management for Beyond 5G Network Coding-enabled Mobile Small Cells, in: Proceedings of the 2nd IEEE 5G World Forum (5GWF), IEEE, Dresden, Germany, 2019. doi:10.1109/5GWF.2019.8911711.

[8] A. Shamir, How to Share a Secret, Communications of the ACM 22 (11) (1979) 612–613. doi:10.1145/359168.359176.

[9] J. K. Liu, M. H. Au, W. Susilo, Self-Generated-Certificate Public Key Cryptography and Certificateless Signature / Encryption Scheme in the Standard Model, in: Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security (ASIACCS), ACM, Singapore, Singapore, 2007, pp. 273–283. doi:10.1145/1229285.1266994.

[10] J. Lai, W. Kou, Self-Generated-Certificate Public Key Encryption Without Pairing, in: T. Okamoto, X. Wang (Eds.), Proceedings of the 10th International Conference on Practice and Theory in Public-Key Cryptography (PKC), Vol. 4450, Springer, Beijing, China, 2007, pp. 476–489. doi:10.1007/978-3-540-71677-8_31.

[11] P. Feldman, A Practical Scheme for Non-interactive Verifiable Secret Sharing, in: Proceedings of the 28th Annual Symposium on Foundations of Computer Science (SFCS), IEEE, Los Angeles, CA, USA, 1987, pp. 427–437. doi:10.1109/SFCS.1987.4.

[12] T. P. Pedersen, Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing, in: J. Feigenbaum (Ed.), Proceedings of the Annual International Cryptology Conference (CRYPTO), Vol. 576, Springer, Santa Barbara, CA, USA, 1991, pp. 129–140. doi:10.1007/3-540-46766-1_9.

[13] A. Herzberg, S. Jarecki, H. Krawczyk, M. Yung, Proactive Secret Sharing Or: How to Cope With Perpetual Leakage, in: D. Coppersmith (Ed.), Proceedings of the Annual International Cryptology Conference (CRYPTO), Springer, Santa Barbara, CA, USA, 1995, pp. 339–352. doi:10.1007/3-540-44750-4_27.

[14] S. Jarecki, Proactive Secret Sharing and Public Key Cryptosystems, Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, USA (1995).

[15] H. Luo, S. Lu, Ubiquitous and Robust Authentication Services for Ad Hoc Wireless Networks, UCLA-CSD-TR-200030, Tech. rep., University of California, Los Angeles, Los Angeles, CA, USA (2000).

[16] H. Luo, J. Kong, P. Zerfos, S. Lu, L. Zhang, URSA: Ubiquitous and Robust Access Control for Mobile Ad Hoc Networks, IEEE/ACM Transactions on Networking 12 (6) (2004) 1049–1063. doi:10.1109/TNET.2004.838598.

[17] M. Girault, Self-Certified Public Keys, in: D. W. Davies (Ed.), Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), Vol. 547, Springer, Brighton, UK, 1991, pp. 490–497. doi:10.1007/3-540-46416-6_42.

[18] H. Deng, D. P. Agrawal, TIDS: Threshold and Identity-Based Security Scheme for Wireless Ad Hoc Networks, Ad Hoc Networks 2 (3) (2004) 291–307. doi:10.1016/j.adhoc.2004.03.005.

[19] E. Da Silva, L. C. P. Albini, Towards a Fully Self-Organized Identity-Based Key Management System for MANETs, in: Proceedings of the 9th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), IEEE, Lyon, France, 2013, pp. 717–723. doi:10.1109/WiMOB.2013.6673435.

[20] Z. Zhang, W. Susilo, R. Raad, Mobile Ad-Hoc Network Key Management with Certificateless Cryptography, in: Proceedings of the 2nd International Conference on Signal Processing and Communication Systems (ICSPCS), IEEE, Gold Coast, QLD, Australia, 2008, pp. 1–10. doi:10.1109/ICSPCS.2008.4813743.

[21] F. Li, M. Shirase, T. Takagi, Key Management Using Certificateless Public Key Cryptography in Ad Hoc Networks, in: J. Cao, M. Li, M.-Y. Wu, J. Chen (Eds.), Proceedings of the 5th IFIP International Conference on Network and Parallel Computing (NPC), Vol. 5245, Springer, Shanghai, China, 2008, pp. 116–126. doi:10.1007/978-3-540-88140-7_11.

[22] M. Gharib, Z. Moradlu, M. A. Doostari, A. Movaghar, Fully Distributed ECC-based Key Management for Mobile Ad Hoc Networks, Computer Networks 113 (C) (2017) 269–283. doi:10.1016/j.comnet.2016.12.017.

[23] J. Lai, W. Kou, K. Chen, Self-Generated-Certificate Public Key Encryption without Pairing and its Application, Information Sciences 181 (11) (2011) 2422–2435. doi:10.1016/j.ins.2011.01.037.

[24] M. Narasimha, G. Tsudik, J. H. Yi, On the Utility of Distributed Cryptography in P2P and MANETs: the Case of Membership Control, in: Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP), IEEE, Atlanta, GA, USA, 2003, pp. 336–345. doi:10.1109/ICNP.2003.1249783.

[25] S. Saeednia, A Note on Girault's Self-Certified Model, Information Processing Letters 86 (6) (2003) 323–327. doi:10.1016/S0020-0190(03)00203-5.

[26] S. S. Al-Riyami, K. G. Paterson, Certificateless Public Key Cryptography, in: C.-S. Laih (Ed.), Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), Vol. 2894, Springer, Taipei, Taiwan, 2003, pp. 452–473. doi:10.1007/978-3-540-40061-5_29.

[27] M. de Ree, G. Mantas, A. Radwan, J. Rodriguez, I. E. Otung, Key Management for Secure Network Coding-enabled Mobile Small Cells, in: V. Sucasas, G. Mantas, S. Althunibat (Eds.), Proceedings of the 9th International Conference on Broadband Communications, Networks, and Systems (BROADNETS), Vol. 263, Springer, Faro, Portugal, 2018, pp. 327–336. doi:10.1007/978-3-030-05195-2_32.

[28] S.-F. Chou, T.-C. Chiu, Y.-J. Yu, A.-C. Pang, Mobile Small Cell Deployment for Next Generation Cellular Networks, in: Proceedings of the 33rd IEEE Global Telecommunications Conference (GLOBECOM), IEEE, Austin, TX, USA, 2014, pp. 4852–4857. doi:10.1109/GLOCOM.2014.7037574.

[29] 3GPP, Enhanced Relays for Energy Efficiency and Extensive Coverage, TR 22.866 V17.0.0, Tech. rep. (2019).

[30] R. Ostrovsky, M. Yung, How To Withstand Mobile Virus Attacks, in: Proceedings of the 10th ACM Symposium on Principles of Distributed

Computing (PODC), ACM, Montreal, QC, Canada, 1991, pp. 51–59. doi:10.1145/112600.112605.
[31] C. P. Schnorr, Efficient Signature Generation by Smart Cards, Journal of Cryptology 4 (3) (1991) 161–174. doi:10.1007/BF00196725.
[32] National Institute of Standards and Technology (NIST), Digital Signature Standard (DSS), FIPS PUB 186-5 (Draft), Tech. rep., Gaithersburg, MD, USA (2019). doi:10.6028/NIST.FIPS.186-5-draft.
[33] T. ElGamal, A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, IEEE Transactions on Information Theory 31 (4) (1985) 469–472. doi:10.1109/TIT.1985.1057074.
[34] M. Bellare, R. Canetti, H. Krawczyk, Keying Hash Functions for Message Authentication, in: N. Koblitz (Ed.), Proceedings of the Annual International Cryptology Conference (CRYPTO), Vol. 1109, Springer, Santa Barbara, CA, USA, 1996, pp. 1–15. doi:10.1007/3-540-68697-5_1.
[35] National Institute of Standards and Technology (NIST), The Keyed-Hash Message Authentication Code (HMAC), FIPS PUB 198-1, Tech. rep., Gaithersburg, MD, USA (2008). doi:10.6028/NIST.FIPS.198-1.
[36] J. Daemen, V. Rijmen, The Design of Rijndael: AES - The Advanced Encryption Standard, Springer, 2002. doi:10.1007/978-3-662-04722-4.
[37] ISO/IEC 19772:2009, Information Technology — Security Techniques — Authenticated Encryption, Tech. rep. (2009).
[38] S. Kent, IP Encapsulating Security Payload (ESP), RFC 4303, Tech. rep. (2005). doi:https://doi.org/10.17487/RFC4303.
[39] K. Hoeper, G. Gong, Bootstrapping Security in Mobile Ad Hoc Networks Using Identity-Based Schemes, in: Y. Xiao, Y. Pan (Eds.), Security in Distributed and Networking Systems, 1st Edition, World Scientific, 2007, Ch. 5, pp. 313–337. doi:10.1142/9789812770103_0013.
[40] N. Saxena, G. Tsudik, J. H. Yi, Efficient Node Admission for Short-Lived Mobile Ad Hoc Networks, in: Proceedings of the 13th IEEE International Conference on Network Protocols (ICNP), IEEE, Boston, MA, USA, 2005, pp. 269–278. doi:10.1109/ICNP.2005.14.
[41] N. Saxena, G. Tsudik, J. H. Yi, Efficient Node Admission and Certificateless Secure Communication in Short-Lived MANETs, IEEE Transactions on Parallel and Distributed Systems 20 (2) (2009) 158–170. doi:10.1109/TPDS.2008.77.
[42] X. Lin, J. G. Andrews, A. Ghosh, R. Ratasuk, An Overview of 3GPP Device-to-Device Proximity Services, IEEE Communications Magazine 52 (4) (2014) 40–48. doi:10.1109/MCOM.2014.6807945.
[43] S. Zhao, R. D. Kent, A. Aggarwal, A Key Management and Secure Routing Integrated Framework for Mobile Ad-Hoc Networks, Ad Hoc Networks 11 (3) (2013) 1046–1061. doi:10.1016/j.adhoc.2012.11.005.