BlocKP: Key Pre-Distribution Based Secure Data Transfer

Mohammed Gharib, Ali Owfi, Fatemeh Afghah, and Elizabeth Serena Bentley

Abstract—Key pre-distribution schemes are promising lightweight solutions to be placed as the cornerstone of key management systems in multi-hop wireless networks. The intermediate decryption-encryption problem, however, is considered as the security threat of such schemes. Multi-path algorithms have been proposed to face such a shortcoming. Alas, these solutions are vulnerable against the node capture attack, where the attacker compromises a fraction of network nodes. In this paper, we propose *BlocKP*, a Blockchain-based solution to increase the resistance of the network against the node capture attack. BlocKP utilizes disjoint key-paths for a key exchange process, where the keying materials form a block at the source-side. Each key-path step generates the next block of the Blockchain until the keying materials reach the destination. BlocKP is a general framework applicable to any key pre-distribution schemes. We propose BlocKP in two versions BlocKP-I and BlocKP-II, where the latter enhance the resistance of BlocKP-I using erasure codes at the cost of negligible control traffic. We analytically show that BlocKP improves the resistance of the network against the node capture attack to almost perfect resistance, using just a small number of paths. We evaluate our solution by performing extensive simulations, considering three baseline key pre-distribution schemes, including probabilistic asymmetric key pre-distribution (PAKP), strong Steiner trade (SST), and unital key pre-distribution (UKP). We equipped these schemes with a compatible multi-path algorithm to offer end-to-end security. Results show that BlocKP improves the throughput up to 5% and decreases the flow completion time into 20% compared to baseline schemes. It has comparable routing traffic, latency, and throughput with augmented solutions but up to 60% improvement in the resistance against the node capture attack.

Index Terms—Secure end-to-end communication, multi-hop wireless network, key pre-distribution, Blockchain.

1 Introduction

Unmanned Terrestrial vehicles (UTVs) and unmanned aerial vehicles (UAVs) are increasingly deployed in a variety of applications such as disaster relief, military missions, terrain reconnaissance, traffic management, and fire detection [1], [2]. UTVs and UAVs, in infrastructure-free environments, use multi-hop cooperative networks as their underlay communication network. The security of the underlying wireless networks is critical in such highly sensitive operations. Alas, the lack of trusted infrastructures and limited node resources make securing communications in such networks challenging. Concretely, while cryptography is a general and powerful approach to improve security, it is not well suited for such infrastructure-free networks. This is because cryptography techniques, such as public key infrastructure (PKI), commonly rely on a key management system, and most of the key management tasks are assigned to a trusted third party (TTP) or several distributed TTPs that are based on infrastructure. In contrast, multi-hop cooperative networks are fully decentralized and lack a fixed infrastructure that can act as the TTP. Plus, nodes in such networks have limited memory, computational, and transmission resources. Consequently, the naive solution of storing all keys in every single node for encrypting and decrypting messages is not practical. The two main reasons are the need for a large amount of storage and processing capability to deal with keying operations in large-scale networks, and the fact that storing stationary keys limits the network from updating them or adding new nodes to the network.

Key pre-distribution schemes seem to be a promising solution due to their distributed and lightweight nature [3]. They store just k keys in each node, where $k \ll n$ and n is the number of network nodes. The set of stored keys in each node is referred to as a *keyring*. Once a node encrypts a message with a key, only those nodes with a shared key are capable of decrypting it. Thus, a pair of nodes can communicate directly and securely if they share a common key. To establish a secure connection between two nodes without a shared key, a key-path has to be found. The key-path is an overlay path in which each pair of adjacent nodes have a secure link between them[1], i.e., they share a common key. To exchange messages, the source initially encrypts its message and forwards it to the first hop on the overlay. The message is then routed over the overlay where each intermediate hop, in turn, decrypts the data, encrypts it again with a key shared with the next hop, and forwards it to the next hop toward the destination.

The intermediate decryption-encryption (DE) steps are an obvious security threat in key pre-distribution schemes [4], [5]. To face this threat, several multi-path solutions have been proposed to design a key-exchange process via multiple disjoint paths [6], [7], [8], [9], [10]. The source and the destination nodes after exchanging some keying materials

M. Gharib, A. Owfi, and F. Afghah are with the Electrical and Computer Engineering Department at Clemson University, Clemson, SC, USA, E. Bentley is with the Air Force Research Lab (AFRL) in Rome, NY, USA, E-mails: {alghari; aowfi; fafghah}@clemson.edu;elizabeth.bentley.3@us.af.mil
DISTRIBUTION STATEMENT A: Approved for Public Release; distribution unlimited AFRL-2021-1447 on 12 May 2021. Other requests shall be referred to AFRL/RIT 525 Brooks Rd Rome, NY 13441.

1. Note that this secure overlay link may span multiple physical nodes, in reality.

via disjoint paths, agree on a key and use it to encrypt and decrypt their communication. The data communication is then encrypted and decrypted at the endpoints, thus no intermediate decryption-encryption steps remain. Although such multi-path algorithms seem to survive the man-in-the-middle attacks, they are still vulnerable against cooperative attacks such as a cooperative version of the man-in-the-middle attack [10]. The attacker can compromise most of the communication by compromising more nodes. We show in this paper that the multi-path solutions have a strong resistance against node capture attack, but only up until a specific node compromise rate of about 10%. The interesting result is that when the rate of compromised nodes surpasses this threshold rate, it causes significant resistance degradation such that the baseline key pre-distribution schemes show better resistance in comparison with the multi-path solutions.

In this paper, we propose BlocKP, a Blockchain-based multi-path key exchange algorithm that improves the resistance of the network against the node capture attack. We propose BlocKP in two versions BlocKP-I and BlocKP-II. We exploit the power of Blockchain in BlocKP-I to improve the resistance using fix-length disjoint paths. The basic idea is to send the keying material via multiple disjoint paths such that the keying material form a block, and this block is recalculated in each hop to form a Blockchain. The destination node, in turn, extracts the keying material from the last block of each Blockchain. The Blockchain helps us to keep the keying materials consistent and guarantees their integrity. To keep the keying material consistent through every Blockchain, we need fix-length disjoint paths. Next, we propose BlocKP-II, a Blockchain-based algorithm which uses erasure coding to improve the resistance of BlocKP-I and relax its assumption of requiring fix-length disjoint paths. We analytically prove the resistance of BlocKP and show how it approaches the perfect resistance, even for a small number of disjoint paths.

The performance of the proposed BlocKP algorithms are evaluated using extensive simulations. First, we evaluate the security strength of BlocKP. Thus, we simulate a mobile ad hoc network with a percentage of compromised nodes. We show that for the network with more than 10% of compromised nodes, BlocKP-I and BloKP-II show about 40% and 60% improvement in the resistance of the network against node capture attack in comparison with the state-of-the-art key exchange solutions. We then simulate a network in ns-2 to evaluate the performance of the proposed algorithm for different number of nodes and different communication loads. In our simulations, we simulate PAKP, SST, and UKP schemes as the baseline key pre-distribution schemes. We augment each of these schemes by a multi-path solution to make their communication secure from end to end. Results show that, in different network setting with different network densities and different network loads, BlocKP decreases the flow completion time up to 20% and improves the network throughput up to 5% in comparison with baseline schemes.

The contribution of this work is to address, for the first time, the common vulnerability problem of the key pre-distribution schemes against the large-scale node capture attack. We propose BlocKP, a Blockchain-based multi-path key exchange solution which i) improves the resistance of mobile ad hoc networks against the node capture attack to more than 30% of captured nodes, in its BlocKP-I version; ii) reduces the number of required disjoint paths by exploiting the power of erasure coding in its BlocKP-II version. We further mathematically analyze the network's resistance against node capture attacks for several state-of-the-art algorithms and both versions of BlocKP algorithms and compare the analytical results with that of simulations. In addition, we exhaustively evaluate the network performance for different combinations of baseline key pre-distribution schemes, the multi-path solutions, and BlocKP under different network load and densities using the ns-2 network simulator.

The rest of this paper is organized as follows. We review the literature work in Section (2). Then, we describe the problem setting along with the adversarial model in Section (3). We describe the proposed BlocKP algorithm in both of its versions in Section (4). We evaluate the security strength and the performance of the network in Sections (5) and (6), respectively. Finally, we conclude the paper in Section (7). The Appendix includes a brief description of elliptic curve cryptography.

2 Related Work

Eschenauer and Gligor [3] proposed the basic idea of key pre-distribution schemes for wireless sensor networks with minimal node resources. Their method was based on a simple idea where several keys are chosen from a key-pool and pre-loaded to each node before the network starts working. Each pair of neighboring nodes can securely communicate if they share a common key. If they are not in the communication range of one another or do not share a common key, a key-path has to be formed between them. The lightweight, fully distributed, and highly connected nature of these schemes attracted the attention of many researchers. However, the initial implementation of key pre-distribution was highly vulnerable against node capture attacks such that the attacker could compromise an entire network's communication by capturing a small number of nodes.

Many researchers proposed extensions to the basic idea of key pre-distribution to improve its security against node capture attack. Chan et al. [11] proposed an algorithm to enable direct communication between adjacent nodes if they share at least q common keys, where the pairwise key is a combination of those keys. Liu et al. [12] proposed to use bivariate polynomials and generate a pairwise key using the polynomial to face the key leakage. Recently, the concept of block design [13] is used to optimize the pre-distribution of the keys. Bechkit et al. [14] proposed Unital key pre-distribution (UKP), an interesting design for key pre-distribution with a high key sharing probability. Ruj et al. [15] proposed another block design based on combinatorial trade, referred to as strong Steiner trade (SST), with much higher resistance against node capture attack but a much lower-key sharing probability.

Gharib et al. [16] proposed a probabilistic asymmetric key pre-distribution (PAKP) scheme. This scheme pre-distributes the public keys of the nodes in an asymmetric

cryptosystem instead of relying on the symmetric system. The idea requires a much higher computational capability for the nodes. Hence, it is not suitable for sensor nodes with extremely limited resources. However, it was proposed for mobile ad hoc networks (MANETs) with higher node capabilities. The interesting property of PAKP is that the pair of nodes do not need to be in the communication range of one another to have a direct secure link [17]. Liu et al. [18] proposed another key pre-distribution scheme based on an asymmetric cryptosystem.

Regardless of the nature of the underlying cryptosystem being symmetric or asymmetric, most of the paths between the pair of source and destination nodes include several intermediate DE steps. Many of the key pre-distribution schemes proposed to consider and enhance only the link security, but not the entire path security [11], [19], [20], [21], [22], [23]. However, the problem of intermediate DE steps, which is the main security threat of entire path security, has been mentioned in a few literature works [4], [7], [8], [9].

The well-known solution to face the intermediate DE threat is to exchange some keying material via multiple disjoint paths. The primary drawback of multi-path algorithms is the imposed traffic caused by these multiple paths. Ling et al. [7] proposed a general multi-path solution for symmetric schemes. They showed that only a small number of disjoint paths are required to improve the resistance against node capture attack and to approach the perfect resistance. Gharib et al. [10] proposed a multi-path solution for asymmetric schemes and showed that the imposed traffic due to the multi-path data transfer not only does not increase the total traffic overhead but also decreases this parameter by solving the path stretch problem. The pair of nodes, which do not have direct secure links, have to use a key-path, which in most cases, is longer than the shortest physical path. This issue is referred to as a path stretch problem. The multi-path key exchange algorithms, after the key exchange process, use the shortest physical path for data transmission. Hence, they prevent data traffic overhead caused by the path stretch.

We show in Section (5) that approaching the perfect resistance against node capture attack by conventional multi-path solutions is only possible for a low fraction node capture rate. Capturing a higher fraction of nodes leads to almost zero resistance even for a large number of disjoint paths. This fact is the main motivation behind developing our proposed BloKP algorithm.

3 Problem setting and Adversarial Model

In this section, we present a formal setting of the network followed by the formal adversarial definition.

3.1 Network Model

We consider a multi-hop wireless network with n nodes, each with a unique ID. All network nodes have the same order of limited resources. Although in some cases, multi-hop networks may have a limited structure, to generalize the problem, we assume an infrastructure-less network without a central network manager. The nodes in the network can be either stationary or mobile, like mobile ad hoc networks (MANET). A traditional routing algorithm is presented to find the shortest path and an arbitrary key pre-distribution scheme is used to secure data transmission between network nodes. Since each node stores just k keys in key pre-distribution and $k << n$, there is no direct secure connection between most of the nodes. However, a chain of consequent direct and secure links establishes each connection. The stored keys in each node are not necessarily stationary. More flexible and practical key update solutions could be considered as described in [24], [25]. However, in any snapshot of time, each node stores k keys.

Assumption 3.1. *We assume that the cryptosystem is secure such that the encrypted message could not be decrypted without the proper key. Hence, a direct secure link cannot be compromised except by capturing the corresponding key.*

Considering Assumption (3.1), the secure data transfer problem could be reduced to compromising the keys. The attacker can compromise the secret keys by compromising the network nodes. Our main goal is to transfer data confidentially among the network nodes and decrease the chance of the attacker to get access to the transferred data by compromising network nodes. We formally define this chance as the complementary probability of the resistance of the network against node capture attack.

Definition 3.2. ***Resistance against node capture attacks*** *is the probability of accessing the transferred data by the attacker when the attacker captures a specific percentage of entire network nodes, uniformly at random. Obviously, the source and the destination nodes are assumed to be not captured.*

We further assume that there is a one-way hash function $h(.)$, which satisfies the minimum conditions of a collision-free function described in [26].

Assumption 3.3. *There is a one-way function h(.), that generates a fixed length hash value for any-length input. We assume that h(.) is collision-free function.*

3.2 Adversarial Model

The main goal of the adversary is to eavesdrop or alter the transferred data between the uncaptured nodes. According to Assumption (3.1), the adversary's goal is reachable only by capturing the corresponding key. Since the secret keys are stored by nodes, the attacker needs to capture nodes . We assume that the adversary can fully capture a fraction of network nodes and get access to their stored keys. The adversary can use the captured nodes to eavesdrop or modify any packet forwarded by them. The adversary can also agree with the captured nodes on specific values or policies. As an instance, the attacker can agree with the captured node on how to perform a man-in-the-middle attack cooperatively to forge a pre-agreed key.

We assume a presence of a simple intrusion detection mechanism capable of detecting the ID-spoofing, i.e. the captured nodes cannot change the ID of the source node. Since the captured nodes are simple network nodes, they have the same limitation in processing and communication capabilities as the honest nodes. Thus, there is no direct and fast communication between the captured nodes such as fiber.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3137900, IEEE Internet of Things Journal

Assumption 3.4. *We assume that the captured node uses the same network to communicate between themselves. Hence, they cannot broadcast a message to all other captured nodes at the same time.*

Assumption 3.5. *In the case of multiple paths between the source and the destination, the source node chooses the path uniformly at random. Hence, the attacker is not able to guess the chosen path.*

4 BlocKP: Secure Data Transfer

BlocKP is, in essence, a three-phase Blockchain-based key exchange algorithm followed by a secure data transfer—the source node generates a request as a block and sends the block via multiple disjoint paths to form multiple Blockchains (Phase 1). The block is updated in each hop and its consistency is checked until reaching the destination (Phase 2). The destination, in turn, validates the blocks and responds to the request such that the source and the destination become able to extract a pairwise key (Phase 3). The data can be then exchanged securely and efficiently between the source and the destination following the shortest path. It is worthy to mention that the utilized Blockchain engages only the participating nodes of disjoint paths in the Blockchain consensus process. In Lemma (4.1) we show that the only possible way for an attacker to get access to transferred data is to perform a man-in-the-middle attack, successfully. By calculating the probability of successful man-in-the-middle attacks, we show that BlocKP improves the resistance against node capture attack for any key pre-distribution schemes, regardless of whether its underlying cryptosystem being symmetric or asymmetric. We propose BlocKP in two different versions, BlocKP-I and BlocKP-II. BlocKP-I has a fairly simpler algorithm and is convenient for a better understanding of the general idea of BlocKP. BlocKP-II, on the other hand, optimizes the resistance of BlocKP-I against node capture attack as well as its performance, by exploiting the power of erasure coding. Table (1) lists the notations that we will use throughout the paper.

Lemma 4.1. *Considering the attacker's goal as eavesdropping or altering the transmitted data, using BlocKP, the only possible way to achieving this goal is to perform a man-in-the-middle attack, successfully. Obviously, by altering the transmitted data, we mean changing the data into other meaningful data.*

Proof. According to Assumption (3.1), the attacker cannot access the encrypted data except by having the corresponding secret key. The true secret key is stored in the source node and the destination node. Hence, the attacker needs to compromise the source node or the destination node to get access to the true key. The only other possibility is to forge a fake key and let the source and the destination nodes believe that this key is the true one. Forging such a fake key is known as man-in-the-middle attack. Hence, the only possible way for the attacker to achieve its goal is to successfully perform a man-in-the-middle attack. □

BlocKP needs a public-private key pair for each node in its key exchange process. Although any asymmetric cryptosystem could be used along with BlocKP, we propose to use elliptic curve cryptography (ECC) for its shorter key length and lower power consumption in comparison with other asymmetric cryptosystems. In our system, each node chooses its own private key and accordingly calculates its corresponding pubic key [2] . Thus, BlocKP does not need any central or trusted party. The use of asymmetric cryptography is limited to signing at most two certificates by the end nodes. Hence, it does not impose a considerable overhead, as we show in Section (6).

TABLE 1
The table of notations.

n	Number of network nodes
k	Size of key-ring
ρ	Sufficient number of vertex-disjoint paths
θ	Threshold for the number of duplicated keys
$P(i)$	Erasure code polynomial in the order of θ
$\mathcal{S}$	Set of collected shares by the destination
$\mathcal{P}$	Set of disjoint paths
x_i	Private key of node i
y_i	Public key of node i
k_{sd}	The source-destination pairwise key
B_i	The i^{th} block in chain
b_i	The information of the i^{th} block
ID_i	Unique ID of node i
$h(.)$	A collision free hash function
p_i	The i^{th} path

4.1 BlocKP-I: Blockchain-based key pre-distribution

The general algorithm of BlocKP is as follows. In the first phase, the source node finds multiple disjoint paths toward the destination and builds a block of request. In the second phase, the block follows disjoint paths toward the destination such that in each hop, the block is updated and the hash value of the new block is attached to the block. The majority consensus of the block is checked in each hop. In the third phase, the destination sends a response encrypted by the source node key, following the shortest physical path. The three-phase algorithm makes the pair of source and destination nodes able to calculate a pairwise key. The three phases are then followed by symmetric data encryption using the calculated pairwise key. The encrypted data, in turn, follows the shortest physical path toward the destination. The details of the three phases are as follows.

Phase 1: The source node finds a set $\mathcal{P}$ that contains ρ number of equal-length vertex disjoint paths $p_i, i = 0, 1, \ldots, \rho - 1$. The source node then forms a request packet to get the destination public key. The request packet is considered as the first block of the Blockchain and represented by B_0. Each block B_i is formed as

$$B_i = (b_i || h(b_i)), \tag{1}$$

where the notation $a||b$ represents the concatenation of a and b,

$$b_i = (i, ID_{src}, ID_{dst}, y_{src}, \mathcal{P}, h(b_{i-1})), \tag{2}$$

y_i is the public key of node i, ID_i is the unique identifier of node i, h is a collision-free hash function, and $h(b_{-1})$ is equal to zero.

Phase 2: For each disjoint path, the source node encrypts the initial block with the shared key of the first intermediate node of that path. The source node then sends each

2. Details are presented in the Appendix.

encrypted packet to the corresponding node. Each intermediate node, in turn, decrypts the packet and forms the next block B_1 according to Equation (1). All the intermediate nodes generate a similar B_1 block. Each intermediate node sends B_1 to the next node in its path. It also sends $h(b_1)$ into all next-hop intermediate nodes, for all other disjoint paths. We show in Lemma (4.2), sending the hash value publicly does not increase the chance of the attacker to get access to the encrypted data. Each intermediate node validates the hash value of the received block and checks the majority consensus of the hash value. In case of inconsistency, the intermediate node requests the block with majority consensus from one of the previous level intermediate nodes. In Lemma (4.3), we show that the previous block could be requested and respond publicly. This procedure is continued until the block reaches the destination. At the first-level intermediate nodes, since the block is coming directly from the source node, the majority consensus of the block is not required to be checked.

Lemma 4.2. *Publicly sending the hash value of the block to the next level intermediate nodes, in the second phase of BlocKP algorithm, does not negatively affect the resistance of the network against node capture attack.*

Proof. To prove this lemma, we consider passive and active attacks, separately. Since the output of hash function is a fixed length random number, eavesdropping on the hash value gives zero information to the attacker. Hence, passive attacks cannot initiate a security risk in this case. For active attacks, we use proof by contradiction. We assume that the captured node changes the hash value to a new value. To initiate a security risk, the new hash value must be a hash value of a similar block including the forged public key[3] instead of the true key. In this case, the intermediate node needs to have the block to be able to change it and calculate the new hash value. Considering that it does not have any knowledge about the communication, e.g. the communication end nodes or the disjoint paths, the captured node does not even able to request the block from the other captured nodes. Considering Assumptions (3.4), other captured nodes cannot instantly broadcast a message between themselves. Further, considering Assumption (3.5), captured nodes even do not know the selected path, hence they cannot deliver the block to the specific intermediate captured node. Hence, the intermediate node cannot access the original block. Thus, this is a contradiction and the intermediate node does not have enough information to be able to generate such a hash value consistent with the forged public key. □

Lemma 4.3. *An intermediate node, in which its received block does not match the majority consensus, can publicly request the block of one of the intermediate nodes with the consensus hash value. The intermediate node, in turn, can also respond publicly. The public request and response in this step do not degrade the resistance of the network against node capture attack.*

Proof. According to Assumption (3.3), the hash function $h(.)$ is collision free. Since the requesting node has a hash value and needs its corresponding block, just the true block can be accepted by the requesting node. Hence, sending them publicly does not degrade the resistance of the network against node capture. □

Phase 3: The destination validates the source node's public key by checking the majority consensus of the received blocks. It encrypts its own public key by the public key of the source node and sends the encrypted key via a shortest physical path. Now, both the source and the destination have the other party's public key and potentially are able to communicate with each other securely. Due to the computational complexity of the asymmetric encryption, we propose to use symmetric encryption for their communication. The pairwise key can be calculated by both the source and the destination as:

$$K_{sd} = x_{src}.y_{dst} = x_{dst}.y_{src}. \tag{3}$$

At this point, the source and the destination can communicate securely via the shortest physical path. Algorithm (1) represents the pseudocode of BlocKP-I. Considering the length of disjoint paths as l, in Lemma (4.4) we show that the attacker needs to compromise at least the majority of the nodes at one of the $l-1$ intermediate levels to become able to capture the communication. In Section (5), we theoretically show that BlocKP-I significantly increases the resistance of the network against node capture attack in comparison with the state-of-the-art.

Lemma 4.4. *In BlocKP-I, the attacker needs to compromise at least the majority of nodes in one intermediate step to be able to forge the public key of the sender and performs the man-in-the-middle attack.*

Proof. We use proof by induction to prove this lemma. First, consider the paths with just one intermediate step, i.e. two hop length. At the intermediate step, since all nodes get their blocks directly from the source node, all nodes has the same block. The captured nodes will forward a forged block with a forged public key. If the attacker captures less than the majority of the nodes, since the destination will check the majority of the blocks, it will choose the right key. Now, assume that the lemma is true for n intermediate steps. Consider a network with $n+1$ intermediate steps, where the attacker could not capture the majority of the nodes in any first n intermediate steps. In the last step of intermediate nodes, all intermediate nodes have the true blocks, getting them directly from the previous step honest node or requested it from them. If the attacker capture less than the majority number of last step intermediate nodes, the destination will receive less than the majority number of forged keys, and it will pick the true key. Hence, in BlocKP-I, the attacker needs to compromise at least the majority of nodes in one intermediate step to be able to forge the public key of the sender and performs man-in-the-middle attack. □

4.2 BlocKP-II: The combination of Blockchain and erasure coding

We propose BlocKP-II to optimize the performance as well as the resistance of BlocKP-I, using erasure coding. Erasure

3. A share of the forged public key in BlocKP-II

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3137900, IEEE Internet of Things Journal

Algorithm 1 BlocKP-I($\mathcal{P}$)

Note: $\mathcal{P}$ is a set of ρ equal size disjoint key-paths from the source node toward the destination.
$j = 0$, {j represents the position of the block in the Blockchain, starting from zero at the source node and ending with the key-path length at the destination node.}
while (j <=key-path length) **do**
  Phase 2:
  if ($j > 0$) **then**
    Receive($e(B_j, p_i(j+1))$),{The intermediate node of the i^{th} path receives an encrypted block from the previous node of its key-path.}
    B_j= decrypt($e(B_j, p_i(j+1)), x_{p_i(j+1)}$), {Each intermediate node decrypts its corresponding block by its private key. }
    validate($h(b_j)$), {Validate the extracted hash value by comparing it to the one received from the previous level nodes.}
    if NOT-VALIDATED **then**
      request(B_j) , {Block B_j is requested from one of the previous level nodes within the majority. }
    end if
  end if
  Phase 3:
  if ($j ==$ key-path length) **then**
    extract(y_{src}), {Extract the source node public key from the validated blocks.}
    send($e(y_{dst}, y_{src})$), {The destination encrypts its own public key by the public key of the source node and sends it to the source node via the shortest physical path.}
    $K_{sd} \leftarrow calculate(y_{src}, y_{dst})$, {Both the source and the destination nodes caluclate the pairwise key using the exchanged public keys.}
  end if
  Phase 1:
  if $j <$ key-path length **then**
    $b_j = (j, ID_{src}, ID_{dst}, y_{src}, \mathcal{P}, h(b_{j-1}))$, {Form the block content.}
    $B_j = (b_j || h(b_j))$, {Attach the hash of the block content to the end of the block.}
    for $(i = 0 : \rho - 1)$ **do**
      $e(B_j, p_i(j+1)) =$ encrypt($B_j, p_i(j+1), y_{p_i(j+1)}$), {Encrypt the block with the public key of the next node in path key-path p_i. }
      send($e(B_j, p_i(j+1))$), {Send the encrypted block to the next node of the key-path p_i. }
      if ($j > 0$) **then**
        send($h(b_j), (\forall k,\ k \neq i)\ p_k(j+1)$), {The node sends $h(b_j)$ to the next nodes of all other paths for validation purposes. }
      end if
    end for
  end if
  j=j+1
end while

coding is, in essence, a method for forwarding error correction in the case of bit erasure, by expanding a message into longer message containing redundant data. The main idea of BlocKP-II is to transfer the public key of the source node to several pieces, say S pieces, which are referred to as key shares. Collecting any θ shares reconstructs the original key. However, any $\theta - 1$ pieces or less do not leak any information about the key. Each share of the key is then sent via a set of disjoint paths similar to the first and the second phases of BlocKP-I. The destination, in turn, collects any θ number of shares to reconstruct the key and proceed with the third phase of BlocKP-I. Accordingly, the three phases of BlocKP-II are as the following. Fig. (1) represents a schematic view of both BlocKP-I (Fig. 1a) and BlocKP-II (Fig. 1b).

Phase 1: The source node chooses $\theta - 1$ random numbers $a_i, i = 1, 2, \ldots, \theta - 1$ and forms the polynomial $P(x)$ as

$$P(x) = y_{src} + \sum_{i=1}^{\theta-1} a_i x^i. \tag{4}$$

$P(0)$ is the public key of the source node and $P(i)$, $i = 1, 2, \ldots, S$ are the key shares. Gathering any θ number of shares leads to reconstructing the public key of the source node.

We use the same number of disjoint paths as BlocKP-I, i.e. ρ. We divide this number into S set of paths $\mathcal{P}_i, i = 0, 1, \ldots, S-1$, each with the ρ_i number of paths such that $\sum_{i=0}^{S-1} \rho_i = \rho$. We name the j^{th} path from i^{th} set $P_{(i,j)}$ where $i = 0, 1, 2, \ldots, S-1$ and $j = 0, 1, 2, \ldots, \rho_i - 1$. The path length in each set has to be constant but different sets may vary in the length of their paths. The source node forms S request packets $B_{(i,0)}$ for $i = 0, 1, \ldots, S-1$. Each block $B_{(i,j)}$ is calculated as

$$B_{(i,j)} = (b_{(i,j)} || h(b_{(i,j)})), \quad \begin{matrix} i = 0, 1, 2, \ldots, S-1, \\ j = 0, 1, \ldots, \rho_i - 1, \end{matrix} \tag{5}$$

where

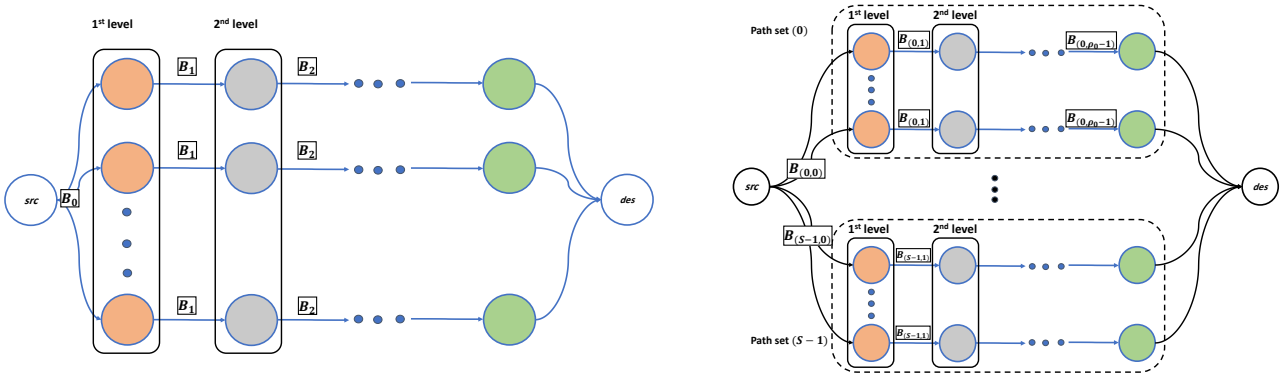
$$b_{(i,j)} = (j, ID_{src}, ID_{dst}, i, \mathcal{P}_i, sign(y_{src}), h(b_{(i,j-1)})), \tag{6}$$

and $h(b_{(i,-1)}) = 0$. The value of $sign(y_{src})$ is the public key of the source node signed by its own private key. This value will be used by the destination node for certifying the extracted public key.

Phase 2: In this phase, the source node encrypts the block $B_{(i,0)}$ by the shared key with the first node of each path and sends the encrypted message to the corresponding node. It is worthy to recall that there are S different blocks $B_{(i,0)}, i = 0, 1, \ldots, S-1$, each one has to be sent via its corresponding set of paths $\mathcal{P}_i$. Each intermediate node, in turn, decrypts the block and generates the next block by increasing the block number and changing the previous hash (Equations 5). It then calculates the current hash value and attaches it to the block (Equation 6).

The intermediate node encrypts the new block by the shared key with the next node and sends the encrypted block to the corresponding node. It needs also to send the hash value to the next-hop intermediate nodes of the other paths of its own path set. Each intermediate node checks the majority consensus of the hash values and steps forward by forming the new block. If the current hash value does

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3137900, IEEE Internet of Things Journal



(a) BlocKP-I (b) BlocKP-II

Fig. 1. A schematic view of BlocKP-I versus BlocKP-II.

not match the majority, then the node requests the previous block from one of the previous level intermediate nodes within the majority. This process will be continued until at least θ sets deliver their blocks to the destination node.

Phase 3: In this phase, the destination collects θ number of key pieces to reconstruct the source node public key. Considering the set of key shares as $\mathcal{S}$, the source node public key could be constructed as

$$y_{src} = \sum_{i \in \mathcal{S}} l_i P(i), \tag{7}$$

where l_i is a Lagrange multiplier at the point zero and could be calculated as

$$l_i = \prod_{j \in \mathcal{S}, j \neq i} \frac{-j}{i-j}. \tag{8}$$

The destination checks the validity of the public key by checking the value of $sign(y_{src})$. The destination then encrypts its own public key by the source node's public key and sends the encrypted message directly to the source node via the shortest physical path. By receiving the destination's key, both the source and the destination nodes can calculate the pairwise key, using Equation (3). The data transfer will then follow the shortest physical path after symmetric encryption, using the pairwise key. The pseudocode of BlocKP-II is represented in Algorithm (2). In Lemma (4.5), we show that the attacker needs to compromise at least the majority of one level in at least θ number of path sets to become able to perform man-in-the-middle attacks and forges its own public key. In Section (5), we show that using BlocKP-II improves the resistance of the network against node capture attack in comparison with BlocKP-I to protect the communications against approximately 10% more compromised nodes.

Lemma 4.5. *In BlocKP-II, the attacker needs to compromise at least the majority of one level in θ number of path sets, to be able to forge the public key of the source node.*

Proof. According to Lagrange interpolation [27], having less than θ point from the polynomial $P(x)$ reveals zero information about the value of that polynomial at point zero, i.e. $P(0)$. Hence, the attacker needs to forge at least θ key shares, to become able to forge a key. To forge each share, according to Lemma (4.4), the attacker needs to compromise at least the majority of the nodes in one intermediate step in the corresponding path set. Hence, the attacker needs to compromise at least a majority of nodes in one step of θ number of path sets to become able to forge a key. □

Algorithm 2 BlocKP-II

$P(x) = y_{src} + \sum_{i=1}^{\theta-1} a_i x^i$, {The source node forms polynomial $P(x)$.}
$\mathcal{P} = findDisjointPaths(src, dst, \rho, S)$, {The source node finds S sets of ρ_i equal size disjoint key-paths toward the destination where $\sum_{i=0}^{S-1} \rho_i = \rho$.}
for $(i = 0 : S-1)$ **do**
  BlocKP-I($\mathcal{P}_i$), {Perform the BlocKP-I algorithm for each path set $\mathcal{P}_i$.}
end for
$P(x)$ =aggregate($\mathcal{S}$), {The destination collects θ shares of the key to form the set $\mathcal{S}$ and reconstruct the polynomial $P(x)$ by aggregating the shares. }
$y_{src} = P(0)$, {The destination extract the source public key.}

5 Security Analysis

In this section, we study the security aspects of baseline key pre-distribution schemes, multi-path solutions, and both versions of BlocKP. We analytically calculate the resistance against the node capture attack and support the analytical calculations by simulations. In our simulation scenarios, we consider a network with 1000 nodes in a 1000×1000 m^2 area, moving according to the random waypoint (RWP) mobility model. Further to RWP, we tested random walk, and Levy-walk mobility models to evaluate the effect of movement patterns on the algorithms. We find that the results are fairly similar for all mobility models. Accordingly, we report only the results of the RWP mobility model.

In our simulations, we considered SST, UKP, and PAKP as the baseline key pre-distribution schemes. Our selection for the schemes is such that we cover different categories of key pre-distributions where PAKP is an asymmetric scheme, SST and UKP are symmetric ones. We further augmented each of the baseline schemes with a multi-path algorithm.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3137900, IEEE Internet of Things Journal

SST and UKP have been augmented by the algorithm of [7] which is proposed for symmetric key pre-distribution schemes. PAKP has been augmented by the algorithm of [10], designed specifically to work based on PAKP. For multi-path solutions, we considered six disjoint paths where we divide them into three sets of paths for BlocKP-II algorithm. We repeated each simulation scenario to reach 0.95 of confidence for the error to be less than 0.01, according to the Monte Carlo theorem [28]. The reported results are the average of all repeated simulation scenarios.

We calculate a general analytical formula for the resistance against node capture attack of multi-path solutions. Lemma (5.1) shows the formula where ρ and l are the number of disjoint paths and path length, respectively. By letting the number of paths equal to one, the formula results in the resistance of baseline key pre-distribution schemes. In this formula, we assume that the attacker compromises a fraction p of the entire network nodes, uniformly at random. Hence, the probability of each node to be compromised is equal to p.

Lemma 5.1. *The resistance of multi-path solutions against node capture attack could be calculated as*

$$R = 1 - (1 - (1 - p)^{l-1})^{\rho}. \tag{9}$$

Proof. In this lemma, we use the reliability analysis proposed in [29]. Each intermediate D-E step is considered as a reliability threat. For a path to be reliable, it should be empty of any compromised node. Hence, the resistance of each path is equal to $(1-p)^{l-1}$, where $1-p$ is the probability of each intermediate node to be uncaptured. We consider the multi-path solutions to work such that all paths have to deliver their key shares to make the destination able to decrypt the data. Thus, the attacker needs to compromise at least an intermediate node from each path to potentially becomes able to perform a successful attack. For ρ disjoint paths, hence, the probability of attacker success when the attacker compromises a fraction p of network nodes is equal to $(1 - (1 - p)^{l-1})^{\rho}$. The resistance of node capture is obviously the complementary probability of attacker access and hence it is equal to

$$R = 1 - (1 - (1 - p)^{l-1})^{\rho}.$$

□

Fig. (2a) shows the results of Equation (9) for a network with 10% of its nodes being compromised. We can conclude from this figure that the multi-path solutions do not need a large number of disjoint paths to reach a high resistance against node capture attack. Since most of the multi-path solutions send only a few number of control packets through the multiple-paths and then use the shortest paths for data transmission, they do not impose serious performance degradation. In Section (6), we show that not only is the overhead of such solutions negligible but also they improve the overall performance by eliminating the path stretch. Fig. (2b and 2c) show the simulation results of baseline schemes and multi-path solutions, respectively, for different fractions of compromised nodes. Considering that the average disjoint path length in our setting is 2.28, 2.42, and 2.58 for PAKP, UKP, and SST, respectively, the simulation results certainly validate our analytical formula. It further shows that although the resistance against node capture attack in multi-path solutions is much higher than that of simple key pre-distribution schemes, after a specific threshold of captured nodes, the resistance of multi-path solutions falls down and becomes worse than that of baseline schemes. However, baseline key pre-distribution schemes represent fairly linear degradation in their resistance by the increment in the number of captured nodes. Lemma (5.2) and (5.3) analytically calculate the resistance of the network against captured nodes for both BlocKP-I and BlocKP-II, respectively.

Lemma 5.2. *The resistance of BlocKP-I algorithm against node capture attack is equal to*

$$R(\rho, l) = \left(1 - \sum_{i=\lfloor \frac{\rho}{2} \rfloor + 1}^{\rho} \binom{\rho}{i} (p^i)(1-p)^{(\rho - i)}\right)^{(l-1)}. \tag{10}$$

We represent this parameter by $R(\rho, l)$ as it depends on the number of vertex-disjoint paths (ρ) and the number of DE steps in each path $(l-1)$.

Proof. According to Lemma (4.4), the attacker to perform a successful attack needs to compromise the majority of the nodes at least at one of the DE steps, in BlocKP-I algorithm. In this case, the attacker can potentially forge its key instead of the true key. Thus, the attacker needs to compromise $\lfloor \frac{\rho}{2} \rfloor + 1$ or $\lfloor \frac{\rho}{2} \rfloor + 2$ or $\ldots$ or ρ nodes in a specific step, where the probability of this event is the summation of binomial mass function for the mentioned cases, i.e. $\sum_{i=\lfloor \frac{\rho}{2} \rfloor + 1}^{\rho} \binom{\rho}{i} (p^i)(1-p)^{(\rho - i)}$. The resistance of one step is then one minus this probability and, hence, the overall resistance is

$$R(\rho, l) = \left(1 - \sum_{i=\lfloor \frac{\rho}{2} \rfloor + 1}^{\rho} \binom{\rho}{i} (p^i)(1-p)^{(\rho - i)}\right)^{(l-1)}.$$

□

Lemma 5.3. *The resistance of BlocKP-II against node capture attack can be calculated as*

$$R = 1 - \left(\sum_{i=\theta}^{S} \binom{S}{i} (1 - R(\rho_i, l_i))^i (R(\rho_i, l_i))^{S-i}\right), \tag{11}$$

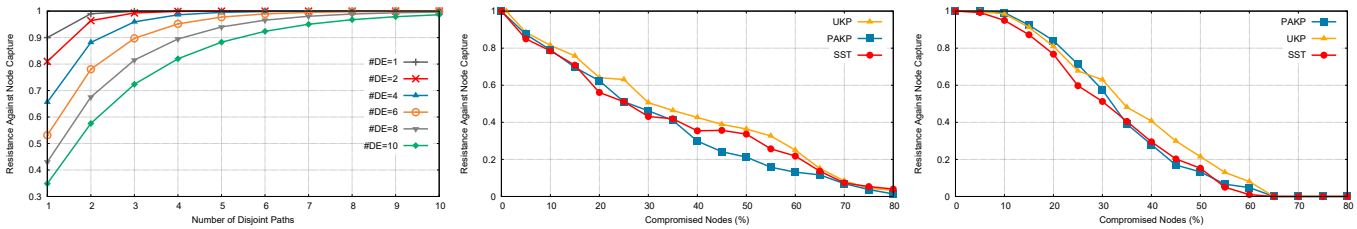
where $R(\rho_i, l_i)$ could be calculated by Equation 10. Considering $\theta = S$, this formula will be reduced to

$$R = 1 - \prod_{i=1}^{S} (1 - R(\rho_i, l_i)).$$

Proof. The resistance against node capture is equal to one minus the probability of attacker success. We know from the proof of Lemma (5.2) that the probability of attacker success in the i^{th} set of paths is $(1 - R(\rho_i, l_i))$, i.e. compromising one share. The attacker needs to successfully compromise at least θ number of shares. Hence, according to binomial distribution this probability is equal to

$$\left(\sum_{i=\theta}^{S} \binom{S}{i} (1 - R(\rho_i, l_i))^i (R(\rho_i, l_i))^{S-i}\right).$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3137900, IEEE Internet of Things Journal



(a) (b) (c)

Fig. 2. A resistance comparison against node capture attack: (a) Analytical results for multi-path solutions with 10% of the nodes being captured; (b) Simulation results for baseline schemes; (c) Simulation results for multi-path solutions.

Since the resistance against node capture is the complementary probability of attacker success, it is equal to

$$R = 1 - \left(\sum_{i=\theta}^{S} \binom{S}{i} (1 - R(\rho_i, l_i))^i (R(\rho_i, l_i))^{S-i} \right).$$

Considering $\theta = S$, the attacker needs to compromise all shares to become able to compromise a connection. Hence, the probability of attacker success is $\prod_{i=1}^{S}(1 - R(\rho_i, l_i))$. Accordingly, the complementary of this probability is the resistance against node capture which is

$$R = 1 - \prod_{i=1}^{S}(1 - R(\rho_i, l_i)).$$

□

Fig. (3) shows the analytical results of resistance against node capture attack in a network with 30% of its nodes captured for BlocKP-I, i.e. Lemma (5.2), and BlocKP-II, i.e. Lemma (5.3). To validate the analytical results and to have a general overview of how powerful the BlocKP algorithm is, Fig. (4) shows the simulation results of the resistance against node capture attack for all the combinations of baseline schemes, multi-path solutions, BlocKP-I, and BlocKP-II. Considering that the number of disjoint paths in these simulations are six, and the average number of intermediate D-E steps are 2.28, 2.42, and 2.58 for PAKP, UKP, and SST, respectively, the results show a certain validation to our analytical calculations, i.e. results of Fig. (3).

6 Performance Evaluation

In this section, we make a comprehensive performance comparison between the different combinations of baseline key pre-distribution algorithms, augmented ones, and both versions of BlocKP. We show that besides the security improvement of BlocKP, its performance is also much higher than that of baseline key pre-distribution schemes and fairly comparable to that of multi-path solutions of the state-of-the-art. We consider throughput, flow completion time (FCT), key-exchange delay, routing traffic, and key exchange overhead as the performance metrics. In this section, we first discuss the simulation setting and follow it by the simulation results and analysis.

6.1 Simulation Setting

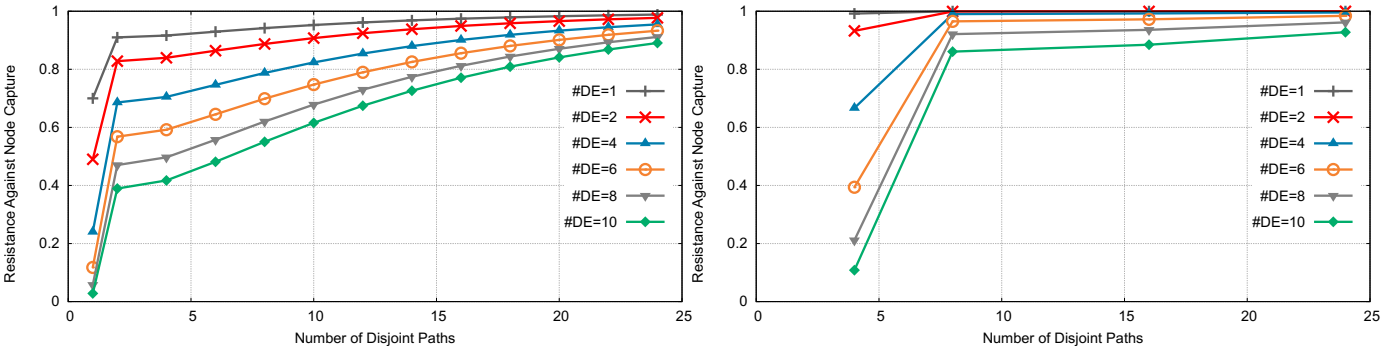
We use ns-2 network simulator [30] to simulate a mobile ad hoc network in a 300×300 m^2 area. To investigate the scalability of different algorithms, we simulate networks starting with 100 nodes and increasing to 200 in increments of 10. In each simulation instance, we choose five pairs of source and destination nodes, uniformly at random. For each pair, a random start time is chosen uniformly in the interval $[0 \quad 60] sec$. Once the set of source-destination pairs is chosen, the same set is used for the simulation of all algorithms. The source node starts to send a file of 5 MB for its corresponding destination using TCP Tahoe on a channel with 1 Mbps bandwidth. To investigate the performance of the network under different loads, in a 100 node setting, we simulate the network for a number of connections ranging from 1 to 10. We find that increasing the number of nodes does not have a major effect on the pattern of the results. Hence, we reported only the results of the network with a fix number of nodes and different numbers of connections. We run each simulation instance for all the combinations of baseline key pre-distribution schemes, including PAKP, SST, and UKP, their augmented version, and their versions supported by both BlocKP-I and BlocKP-II. We use AODV as the underlay routing protocol. The nodes are assumed to be mobile following random waypoint (RWP), random walk (RW), and Levy walk (LW) mobility models. We find that the results show the same pattern for all the mentioned models. Thus we represent only the results of the RWP model in this paper. A distance model with a communication range of 100 m is considered for all nodes.

For the key pre-distribution process, we consider PAKP, SST, and UKP schemes. In SST, the probability of storing a shared key between any pair of nodes does not exceed 0.25, which leads to longer key path length. However, the attacker needs to compromise more nodes to be able to access all keys. In 2-UKP, the probability of storing a shared key is much higher, which leads to shorter key paths, but the attacker can get access to all distributed keys by compromising a small numbers of nodes. Each node stores ten keys, and six disjoint paths are considered for multi-path solutions. For BlocKP-II we divide the set of paths into three subsets, each with two disjoint paths.

6.2 Simulation Result

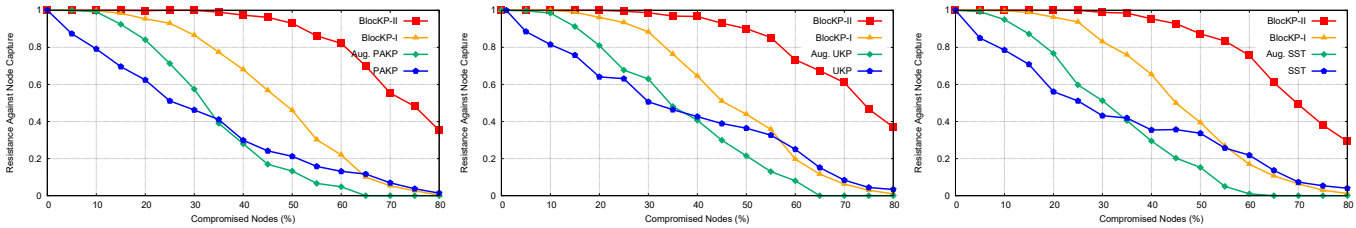
In this section, we investigate the network throughput, flow completion time, and control traffic as the performance metrics. We further compare the delay and control traffic of the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3137900, IEEE Internet of Things Journal



(a) BlocKP-I (b) BlocKP-II

Fig. 3. A comparison of the resistance against 30% of the nodes in a network being captured (analytical results).



(a) PAKP (b) UKP (c) SST

Fig. 4. A general comparison of resistance against node capture attack among baseline schemes, multi-path solutions, and both versions of BlocKP algorithms (simulation results).

key exchange process for those algorithms that include such a step. We repeat each simulation scenario 20 times, and the presented results are the average of all simulated scenarios. We measure the throughput as the ratio of successful packet delivery over the bandwidth. Fig. (5) shows this metric for different schemes and their augmented algorithms. Since the simple baseline schemes suffer from path stretch problem, they show lower throughput in comparison with the multi-path solutions. In multi-path solutions, the data transfer process follows the shortest path between the source and the destination, after a key-exchange process.

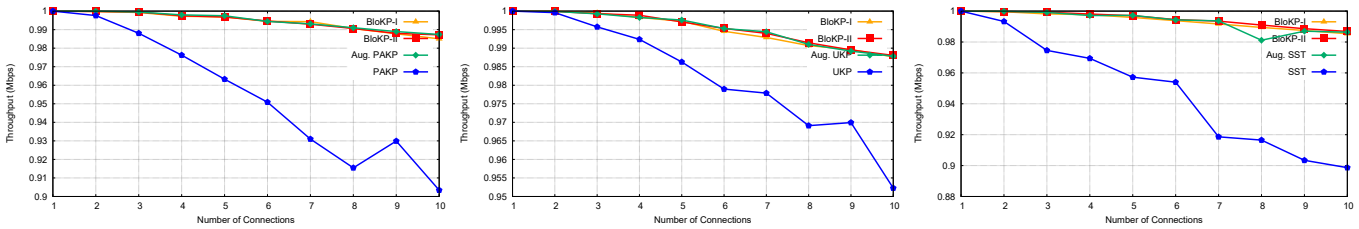
The next measured metric is flow completion time. We measure this metric as the time between the first packet sent by the source node and the last packet of the 5 MB file received by the destination. This time includes the time required for the key-exchange process in multi-path algorithms. Fig. (6) shows the results of FCT for different baseline schemes and their multi-path extensions. Since the increment in the number of connections leads to more congestion, the FCT increases by the increment in the number of concurrent connections. Again, the baseline key pre-distribution schemes have longer FCT due to the longer key paths.

Fig. (7) shows only the time consumed for the key-exchange process for different multi-path solutions. Aug. UKP and aug. SST algorithms have just one phase in their key exchange process. These algorithms send different key pieces via disjoint paths and then immediately start sending the data packets. The destination becomes able to decrypt data packets after receiving all key pieces. Accordingly, we calculate the time required for the key exchange process as the time starting from the transfer of the first key exchange packet by the source node until receiving the last piece of the key by the destination. This parameter for aug. PAKP, since it includes one more phase, is calculated as the time between sending the first key exchange packet and receiving the destination key by the source node. While aug. UKP and aug. SST show the shortest key exchange process, BlocKP-I generally outperforms BlocKP-II for all algorithms. However, BloKP-I needs a set of same length disjoint paths, which in some cases might not be available. Thus, we calculated the key exchange failure ratio as the ratio of unavailability of enough number of fixed length disjoint paths. Table (2) shows the failure ratio for all multi-path solutions. This table shows that even for the SST scheme, which suffers from low key sharing probability, BlocKP-II failure rate approaches to zero. However, BlocKP-I suffers from a higher failure rate in comparison with other solutions. Since in the PAKP scheme, the key paths have to be disjoint only in the overlay, a lower failure rate could be seen.

It is well known that in dynamic networks, the amount of control traffic is one of the main performance evaluation metrics. This metric includes routing traffic, key exchange traffic, and so on. We measured the average control traffic per each connection for different numbers of connections. Fig. (8) shows the results. By the increment in the number of concurrent connections, we see an obvious increment in the control traffic of the baseline key pre-distribution schemes. The main reason is the longer path length in comparison with multi-path solutions. BlocKP-I shows slightly higher traffic volume due to its overall longer key-path length.

We further measured the control traffic of the key exchange process for multi-path solutions. Fig. (9) shows the results. The high key sharing probability in the UKP scheme,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3137900, IEEE Internet of Things Journal

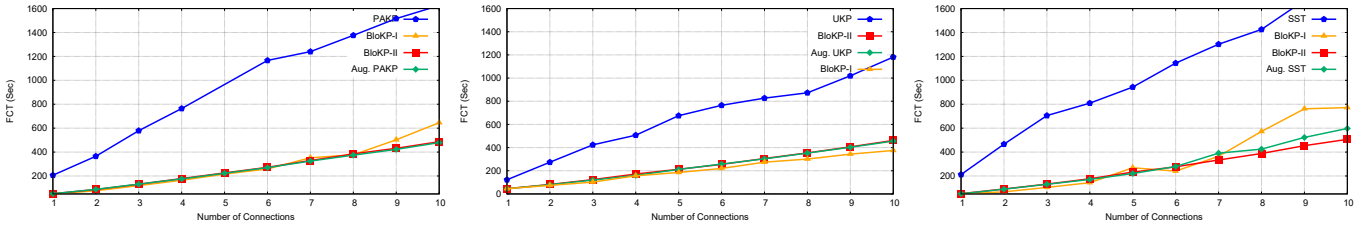


(a) PAKP (b) UKP (c) SST

Fig. 5. A comparison of the throughput for different key pre-distribution schemes.

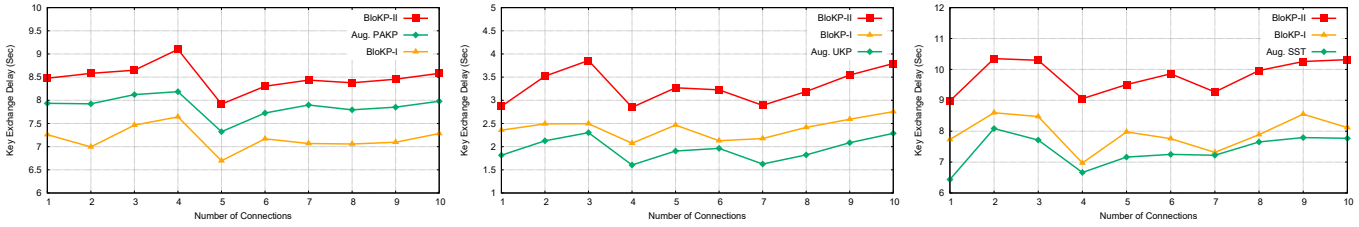
TABLE 2
Failure ratio of the multi-path algorithms.

Scheme	PAKP			SST			UKP		
Algorithm	Aug.	BloKP-I	BlocKP-II	Aug.	BloKP-I	BlocKP-II	Aug.	BloKP-I	BlocKP-II
Failure Rate	0.0	0.117	0.003	0.115	0.240	0.003	0.0	0.168	0.0



(a) PAKP (b) UKP (c) SST

Fig. 6. A comparison of the flow completion time for different key pre-distribution schemes.



(a) PAKP (b) UKP (c) SST

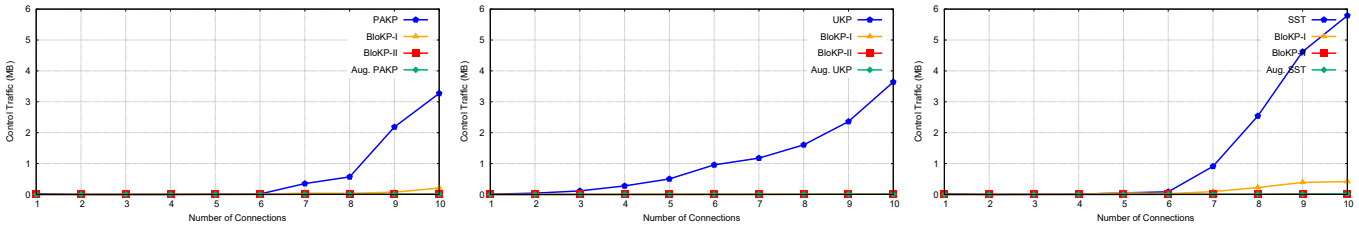
Fig. 7. A comparison of the key exchange delay for different key pre-distribution schemes.

which results in shorter paths along with the lightweight nature of the Aug. UKP key exchange process, leads this scheme to have relatively less control traffic in its key exchange process. The low key sharing probability in SST, in turn, leads this scheme to experience more control traffic. Generally, BlokcKP-II generates a lower volume of control traffic in comparison with BlocKP-I, due to its lower key-path length.

7 CONCLUSION

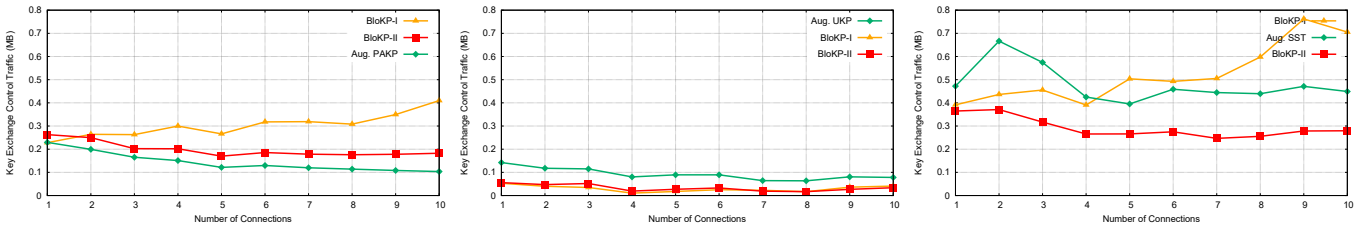
Intermediate D-E problem caused by key pre-distribution schemes is a dire security threat where the multi-path algorithms were known as effective solutions. In this paper, we analytically showed that although the multi-path solutions are effective, they are vulnerable against a high ratio of node capture. We supported and validated our analytical results with simulations. To face this deficiency, we proposed the BlocKP-I algorithm based on Blockchain and improved its idea by exploiting the power of erasure coding to propose BlocKP-II. We analytically showed that BlocKP could resist the high ratio of node capture, even for more than 50% of entire network nodes. We further performed exhaustive simulations to show the performance of BlocKP in both versions. We showed that BlocKP improves the network performance in comparison with the baseline key pre-distribution schemes, and it has a comparable performance with the literature multi-path algorithms. Since energy consumption is crucial in multi-hop networks, we suggest a comprehensive power evaluation for the key pre-distribution algorithms, as future work. As another future direction, we recommend implementing the BlocKP algorithm in the real world. Designing different attack scenarios to evaluate the sensitivity of the key pre-distribution-based algorithm is also of paramount importance.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3137900, IEEE Internet of Things Journal



(a) PAKP (b) UKP (c) SST

Fig. 8. A comparison of the end-to-end control traffic for different key pre-distribution schemes.



(a) PAKP (b) UKP (c) SST

Fig. 9. A comparison of the key exchange control traffic for different key pre-distribution schemes.

8 Acknowledgment

This material is based upon the work supported by the National Science Foundation under Grant No. 1827753.

Appendix

Elliptic curve cryptography (ECC) is well-known as the asymmetric cryptography algorithm with the lowest computational complexity and shortest key length for the same level of security strength compared to other asymmetric algorithms [31], [32]. In ECC, similar to other asymmetric algorithms, each node has a pair of public and private keys. Other key system parameters include (p, a, b, G, n, h). An elliptic curve has to be defined as a plane curve over a finite field $\mathbb{F}_p$, which satisfies

$$y^2 = x^3 + ax + b. \tag{12}$$

While G is the base point, n is the smallest positive number which satisfies $nG = \mathcal{O}$, where $\mathcal{O}$ is a point at infinity. Finally, h is a small integer number $h = \frac{1}{n}|E(\mathbb{F}_p)|$. In this system, the private key is a random integer number chosen from the interval $[1 \quad n-1]$. Considering the private key as x, the corresponding public key is $y = x.G$, which is resulted from x times adding G to itself.

References

[1] X. Tan, Y. Jin, W. Feng, S. Wang, and Y. Yang, "Scheduling of distributed collaborative tasks on ndn based manet," in *Proceedings of the ACM SIGCOMM 2019 Workshop on Mobile AirGround Edge Computing, Systems, Networks, and Applications*, ser. MAGESys'19. New York, NY, USA: Association for Computing Machinery, 2019, p. 36–42.

[2] W. Mei and R. Zhang, "Cooperative downlink interference transmission and cancellation for cellular-connected uav: A divide-and-conquer approach," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1297–1311, 2020.

[3] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, ser. CCS '02. New York, NY, USA: ACM, 2002, pp. 41–47.

[4] Yun Zhou, Yanchao Zhang, and Yuguang Fang, "Llk: a link-layer key establishment scheme for wireless sensor networks," in *IEEE Wireless Communications and Networking Conference, 2005*, vol. 4, March 2005, pp. 1921–1926 Vol. 4.

[5] M. Gharib, H. Yousefi'zadeh, and A. Movaghar, "Secure overlay routing using key pre-distribution: A linear distance optimization approach," *IEEE Transactions on Mobile Computing*, vol. 15, no. 9, pp. 2333–2344, 2016.

[6] P. Papadimitratos and Z. J. Haas, "Secure message transmission in mobile ad hoc networks," *Ad Hoc Networks*, vol. 1, no. 1, pp. 193 – 209, 2003.

[7] Hui Ling and T. Znati, "End-to-end pairwise key establishment using multi-path in wireless sensor network," in *GLOBECOM '05. IEEE Global Telecommunications Conference, 2005.*, vol. 3, Nov 2005, pp. 5 pp.–.

[8] G. Li, H. Ling, and T. Znati, "Path key establishment using multiple secured paths in wireless sensor networks," in *Proceedings of the 2005 ACM Conference on Emerging Network Experiment and Technology*, ser. CoNEXT '05. New York, NY, USA: ACM, 2005, pp. 43–49.

[9] J.-P. Sheu and J.-C. Cheng, "Pair-wise path key establishment in wireless sensor networks," *Computer Communications*, vol. 30, no. 11, pp. 2365 – 2374, 2007, special issue on security on wireless ad hoc and sensor networks.

[10] M. Gharib, A. Owfi, and S. Ghorbani, "Kpsec: Secure end-to-end communications for multi-hop wireless networks," 2019.

[11] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, May 2003, pp. 197–213.

[12] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security*, ser. CCS '03. New York, NY, USA: ACM, 2003, pp. 52–61.

[13] E. F. Assmus and J. D. Key, *Designs and their codes*. Cambridge University Press, 1992.

[14] W. Bechkit, Y. Challal, A. Bouabdallah, and V. Tarokh, "A highly scalable key pre-distribution scheme for wireless sensor networks," *Wireless Communications, IEEE Transactions on*, vol. 12, no. 2, pp. 948–959, February 2013.

[15] S. Ruj, A. Nayak, and I. Stojmenovic, "Fully secure pairwise and triple key distribution in wireless sensor networks using combina-

torial designs," in *INFOCOM, 2011 Proceedings IEEE*, April 2011, pp. 326–330.
[16] M. Gharib, E. Emamjomeh-Zadeh, A. Norouzi-Fard, and A. Movaghar, "A novel probabilistic key management algorithm for large-scale manets," in *2013 27th International Conference on Advanced Information Networking and Applications Workshops*, March 2013, pp. 349–356.
[17] M. Gharib, H. Yousefi'zadeh, and A. Movaghar, "Secure overlay routing for large scale networks," *IEEE Transactions on Network Science and Engineering*, pp. 1–1, 2018.
[18] Z. Liu, J. Ma, Q. Huang, and S. Moon, "Asymmetric key predistribution scheme for sensor networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1366–1372, March 2009.
[19] J. Zhao, "On resilience and connectivity of secure wireless sensor networks under node capture attacks," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 3, pp. 557–571, March 2017.
[20] ——, "Topological properties of secure wireless sensor networks under the q -composite key predistribution scheme with unreliable links," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1789–1802, June 2017.
[21] F. Gandino, R. Ferrero, and M. Rebaudengo, "A key distribution scheme for mobile wireless sensor networks: q - s -composite," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 34–47, Jan 2017.
[22] O. Yagan and A. M. Makowski, "Wireless sensor networks under the random pairwise key predistribution scheme: Can resiliency be achieved with small key rings?" *IEEE/ACM Transactions on Networking*, vol. 24, no. 6, pp. 3383–3396, December 2016.
[23] M. Gharib, H. Yousefi'zaddeh, and A. Movaghar, "A survey of key pre-distribution and overlay routing in unstructured wireless networks," *Transactions on Computer Science & Engineering and Electrical Engineering*, vol. 23, no. 6, pp. 2831–2844, December 2016.
[24] S. Capkun, J. P. Hubaux, and L. Buttyan, "Mobility helps peer-to-peer security," *IEEE Transactions on Mobile Computing*, vol. 5, no. 1, pp. 43–51, Jan 2006.
[25] M. Gharib, M. Minaei, M. Golkari, and A. Movaghar, "Expert key selection impact on the manets' performance using probabilistic key management algorithm," in *Proceedings of the 6th International Conference on Security of Information and Networks*, ser. SIN '13. New York, NY, USA: ACM, 2013, pp. 347–351. [Online]. Available: http://doi.acm.org/10.1145/2523514.2523556
[26] A. Russell, "Necessary and sufficient conditions for collision-free hashing," *Journal of Cryptology: The Journal of the International Association for Cryptologie Research (IACR)*, vol. 8, no. 2, pp. 87–99, 1995, cited By 17.
[27] M. Hazewinkel, *Lagrange interpolation formula, Encyclopedia of Mathematics*. Springer Science+Business Media B.V. / Kluwer Academic Publishers, 2001.
[28] M. H. Kalos and P. A. Whitlock, *Monte Carlo Methods*. WILEY-VCH, 2008.
[29] K. Trivedi, *Probability and Statistics with Reliability, Queuitgo and Computer Science Applications*. Wiley-Interscience Publication, 2002.
[30] (2020, Sep.). [Online]. Available: http://www.isi.edu/nsnam/ns/
[31] *Standards for efficient cryptography, SEC 1: Elliptic Curve Cryptography*. Certicom Research, 2009.
[32] M. Gharib, Z. Moradlou, M. A. Doostari, and A. Movaghar, "Fully distributed ecc-based key management for mobile ad hoc networks," *Computer Networks*, vol. 113, pp. 269 – 283, 2017.

Mohammed Gharib is currently a research scholar at the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, US. He had a postdoctoral research fellow position at the School of Informatics, Computing, and Cyber Systems, Northern Arizona University, Flagstaff, Arizona, US and the Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA. He was a senior researcher at the School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran. He was also a senior lecturer at the Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. Mohammed received a BS degree from the Department of Computer Engineering, Baghdad University of Technology, Baghdad, Iraq, and both MS and Ph.D. degrees from the Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. Mohammed had visited the University of California, Irvine, and Maraldalen University, Sweden. His main research interest is the performance evaluation of computer networks, including wireless networks, cloud networks, data center networking, and their security aspects..

Ali Owfi is currently a PhD student at the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, US. His research interest include ad hoc-networks and decentralized networking.

Fatemeh Afghah is an Associate Professor with the Electrical and Computer Engineering Department at Clemson University. Prior to joining Clemson University, she was an Associate Professor with the School of Informatics, Computing and Cyber Systems, Northern Arizona University, where she was the Director of Wireless Networking and Information Processing (WiNIP) Laboratory. Her research interests include wireless communication networks, decision making in multi-agent systems, radio spectrum management, UAV networks, security and artificial intelligence in healthcare. Her recent project involves autonomous decision making in uncertain environments, using autonomous vehicles for disaster management and IoT security. She is the recipient of several awards including the Air Force Office of Scientific Research Young Investigator Award in 2019, NSF CAREER Award in 2020, NAU's Most Promising New Scholar Award in 2020, and NSF CISE Research Initiation Initiative (CRII) Award in 2017. She is the author/co-author of over 100 peer-reviewed publications and served as the associate editor for several journals including Elsevier Journal of Network and Computer Applications, Ad hoc networks, Computer Networks, Springer Neural Processing Letters and the organizer and TPC chair for several international IEEE workshops in the field of UAV communications and AI, including IEEE INFOCOM Workshop on Wireless Sensor, Robot, and UAV Networks (WiSRAN'19), IEEE WOWMOM Workshop on Wireless Networking, Planning, and Computing for UAV Swarms (SwarmNet'20&21), and 2021 NSF Smart Health PI workshop on "Smart Health in the AI and COVID Era".

Elizabeth Serena Bentley has a B.S. degree in Electrical Engineering from Cornell University, a M.S. degree in Electrical Engineering from Lehigh University, and a Ph.D. degree in Electrical Engineering from University at Buffalo. She was a National Research Council Post-Doctoral Research Associate at the Air Force Research Laboratory (AFRL) in Rome, NY. Currently, she is employed by the AFRL Information Directorate, performing in-house research and development in the Communication Technology & Systems Branch and managing the Cross-Layer Heterogeneous Autonomous Resilient On-Demand Networks (CHARON) program that focuses on mission-responsive swarm networking. Her research interests are in cross-layer optimization, swarm networking, directional networking, wireless multiple-access communications, and modeling and simulation.