A Novel Multi-Path Anonymous Randomized Key Distribution Scheme for Geo Distributed Networks

Ashish Nanda[1], Priyadarsi Nanda[1], Mohammed S. Obaidat (*Fellow of IEEE*)[2], Xiangjian He[1], and Deepak Puthal[3]
[1] Faculty of Engineering and IT, University of Technology, Sydney, Australia
[2] King Abdullah II School of Information Technology, The University of Jordan, Amman, Jordan
[2] ECE Department, Nazarbayev University, Astana, Kazakhstan; and University of Science and Technology Beijing, China
[3] School of Computing, Newcastle University, UK
Email: {dr.Ashish.Nanda, Msobaidat, Deepak.Puthal}@gmail.com; {Priyadarsi.Nanda, Xiangjian.He}@uts.edu.au

***Abstract*— A major concern in distributed networks is the ability to provide acceptable levels of security. This is achieved by using encryption and authentication mechanisms that depend on encryption keys. However, given the ever-expanding nature of the network, it is difficult to keep setting up authorities that can aid the key-exchange process. This paper presents a novel solution to the challenge of exchanging keys of a large, distributed network without the need to set up additional authorities. The key-exchange scheme presented takes advantage of features such as packet anonymity, random selection and a multi-path approach for the exchange process. The paper also discusses the effectiveness of the proposed scheme against various threat scenarios.**

Keywords—Key Distribution; Secret Key; multipath anonymous randomized key distribution, pairwise key exchange

I. Introduction

As distributed networks are decentralized, they require a robust scheme to preserve the security of the devices connected to them. In the last decade, an increasing number of devices has become a part of such networks and the trend is still growing. One of the main reasons for their increased popularity is the fact that they can be set up anywhere with minimal infrastructure requirements. However, with an increase in unattended standalone networks in adversarial environments, the need for improved security is very high [1, 4]. In this paper, we present a new key-distribution scheme, specifically designed for distributed networks.

Although the security of such networks is easily improved using end-to-end encryption, the most crucial component of the process is the key exchange as it provides the key to be used for encryption/decryption process. The key-exchange can be attempted using various schemes; each one with its own set of keying style [7] as depicted in Table I. The approach to be used for a specific network is determined after analyzing the various components and configuration of that distributed network.

The encryption/decryption process can be conducted using a symmetric key where a master key is distributed throughout the network. However, if even one of the devices in the network is compromised, it will compromise the whole network [2]. To avoid this, individual asymmetric or public-private key pairs are preferred as the public key can only be used to encrypt and only its linked private key can decrypt. This way even if a device is compromised, it would not be able to compromise the whole network.

TABLE I. Types of key distribution approaches

Approach	Mechanism	Keying Style
Probabilistic	Pre - Distribution	Random-Key
		Pair-Wise key
Deterministic	Pre – Distribution	Pair-Wise key
		Combinatorial
	Dynamic Key Generation	Master Key
		Key Matrix
		Polynomial
Hybrid	Pre – Distribution	Combinatorial
	Dynamic Key Generation	Key Matrix
		Polynomial

These keys (or keying material) can be pre-loaded onto the devices (key pre-distribution) and used when connecting to the network. This approach is however very static and can disrupt the network by compromising enough devices to be able to compromise other keys being distributed [3]. To avoid such scenarios, dynamic key-pairs are used and can be updated regularly to maintain a higher level of security [5, 6]. In addition to the approach and keying style, there are certain constraints, which need to be considered when designing a key distribution scheme for distributed networks [8].

The devices that form a distributed network can span over very large areas and hence communicate using the concept of hopping. This implies that any data sent over the network will travel through other devices before finally reaching its destination. The distributed networks also contain mobile nodes that keep changing locations, and in doing so, also change the network layout [10]. A major factor on which the key strength and size depend is the computational resources available with the devices in the networks [9].

To address the above concerns and constraints, we present out proposed Multi-Path Anonymous Randomized Key distribution scheme. Section II discusses the assumptions made regarding the applicable network structure and provides the notations used in this article. The paper then presents the Multi-Path Anonymous Randomized Key (MPARK) distribution scheme and its components in Section III. In Section IV, we expand upon the various security threats and show how they are overcome using our scheme. Finally, we conclude our proposed scheme as well as discuss its future development in Section V.

II. Related Studies and Motivation

The network uses different types of crypto keys to perform encryption and can also be used for authentication purposes. Hence, the crypto keys play an important part in the security framework by strengthening other components of the framework. Key management techniques for any network can be divided into two main components: the key generation process and the key distribution process. These two processes work independently of each other but contribute equally to the strength of the network.

Key Generation: The key generation process is designated to create the crypto keys that will be used by the device for encryption and in certain cases authentication. The crypto keys can be generated using various techniques depending upon the requirement. As the crypto keys are of two types, symmetric (secret) or asymmetric (public-private), the technique used must be selected accordingly. For each secret key or public-private key pair generated, there are certain factors that govern the strength provided by the keys when in operation. These factors include the followings.

Seed Value: The seed value in an initial string of characters is used to create the key. The randomness of the characters in a seed value determines the strength of the key. This randomness can be true randomness captured from a source such as the clicks from a Geiger Counter [12], using the voice or thermal noise captured by sound equipment [13] or using a light source. The randomness can also be generated using pseudorandom sources that use certain algorithms to generate almost random numbers. Although true random sources and considered better than pseudorandom sources, the random sources have, at any given time, collected only a finite set of available characters as compared to the infinite number of characters that a pseudorandom generator can provide anytime.

Key Length: The length of the key is also considered important as it defines the number of characters available to the encryption mechanism to use for encryption. The more characters a key has, the greater its length and the greater the time it would take to guess the combination. This is because the number of possible combinations increases exponentially with the increase in several characters used as well as the type of characters used. For example, a 2-character binary-based key will have 4 possible combinations as compared to a 10-character integer-based key that can have 10,000,000,000 possible combinations, thus, increasing the complexity and thereby utilizing more time to calculate.

Key Distribution: The key distribution process involves the exchange of newly generated crypto keys amongst devices and any possible central entities on the network. The key exchange is a crucial stage as any compromise or tampering with the keys can sabotage the other components of the security framework. The key exchange involves setting up a communication channel with a device or central entity in order to send the crypto key. The type of key being exchanged defines the type of security required for the transmission channel. A secret key requires a secure connection or a trusted party connection to be exchanged safely as it can be used for both the encryption and decryption process, whereas a public-private key pair can use an open channel as only the public key needs to be exchanged which can only be used to encrypt the data. Various existing schemes present different ways of achieving the same [14, 15] using a certification authority responsible for storing and distributing keys, hierarchical based key distribution schemes, one-way tree-based distribution, multi-key distribution, among others.

III. Assumptions and Notations

This section states some realistic assumptions for the distributed network scenario where our proposed key distribution scheme MPARK can be applied. This section also defines the notations used while defining the proposed scheme.

A. *Network Assumptions.*

The MPARK distribution scheme considers some realistic assumptions to maintain an adequate level of security. As it is an integral part of a network, there must exist a central entity referred to as a server. The server must be able to keep a record of all the public keys in use and the devices they are linked to. The server must also have enough processing power to conduct complex calculations and fast storage to traverse and lookup data quickly. The server must always be run and maintained as an active connection to the network devices directly or through proxy devices.

Another integral part of a distributed network is the devices it is formed of; hence it is important that all the devices have adequate resources. The devices must have enough processing power to be able to generate their own asymmetric key pairs (used in key exchange) and symmetric key (used as a session key). In addition to adequate processing power, they must also contain enough memory to store a few megabytes of data such as the key pool and keying material.

A new device that is connecting or seeking a connection to the network must be authenticated before the key exchange is initiated. Until the authentication process is completed, the new device must not be given full or partial access to the network. In order to maintain the integrity of the proposed scheme, it is also important that the new device is within range of two or more edge devices (a device which is already a part of the network) before the key exchange is initiated.

B. *Notations used*

Table II. lists the various notations used, the component they refer to and their descriptions. These notations are used to refer to integral components of the proposed scheme.

IV. Proposed Key Distribution Scheme

This section presents the proposed MPARK distribution scheme to address the issues stated in the above section. The aim is to use anonymity and randomness to further reduce the risks involved in the key exchange over an insecure wireless channel in distributed systems. This section discusses the working of the MPARK distribution scheme, its components, the details of the methodology used and its effectiveness.

The proposed scheme has been divided into its major components to provide its key details. Each component provides its unique features and when combined all together, form a strong key distribution scheme. The major components are individually discussed below.

TABLE II. LIST OF NOTATIONS USED.

Notations	Component	Description
D_N	New Device	A new user device that is attempting a key exchange.
D_E	Edge Device	A user device which is currently part of the network and has an existent key pair set up with the server
Server	Central Server	A centralized system used to keep a record of public keys and the devices they correspond to. A user device can use it to look up the public key for another device it wishes to contact.
K_{PU}	Public Key	A unique key linked to a user device's private key. It is publicly stored on the server and can be used by any device only to encrypt data destined for its user device
K_{PI}	Private Key	A unique secret key stored securely on the user device. Only a specific private key can be used to decrypt data that was encrypted with its linked public key.
K_P	Key Pair	A set of Private Key and Public Key that are linked to each other.
Pkt	Data Packet	A normal TCP/IP packet used to encapsulate data and other important information while it travels from the source device to the destination device.
TTL	Time To Live	A time value which represents the validity of the data packet. If the time to live is expired, the data packet will be discarded.
CON	Active Connections	An active connection refers to an ongoing connection between a device and its in-range neighbor device.
K_{POOL}	Pool of Keys	A collection of Public Keys / Private Keys
IV	Initialization Vector	A random starting variable used as an initializer for the encryption process.
N_P	Number of K_P	The total number of key pairs created during a single key distribution process
N_E	Number of D_E	The number of edge devices involved in the Key Distribution process.
N_S	A set of K_P	The set of Key Pairs (Can also refer to Public Keys in general) that are part of a single set of Keys packed in a data packet
Rnd	Randomizer	An absolute (or pseudo) randomness generator
S_V	Seed Value	The initial elements used as a reference when creating key pairs.

A. Initial contact

The initial contact can be defined as the moment a new device is being added or has requested to be added to the network. During this phase, the new device undergoes the authentication process. The steps involved in the authentication process will vary depending upon the type of network being addressed and the various constraints it uses. In either scenario, the authentication must be completed successfully before a device is provided access to the network.

Only after the authentication process is successfully completed, the new device is provided with a public key K_{PU} (Fig. 2) to communicate with the network server using a secure line. Before the key-pairs K_P are generated, the new device also calculates a list of possible edge nodes that can be used to initiate the key exchange process. This list of edge devices N_E is used while generating encryption keys and is defined further in the next section.

B. Key-Pair generation

The key generation process is usually an integral part of the scheme as the strength of the key is directly related to the amount of time it will take to break it. It also depends upon the type of devices being used in the network as the key length will be selected based on the computation possible with onboard resources. As in a distributed network, the data travels using hops, the size of the encrypted data is also an important factor when selecting an encryption algorithm [11].

Our scheme uses asymmetric keys, so only the public key K_{PU} is exchanged, which is unique for each device on the network. In addition, as the key pair K_P is generated by the device itself, the private key K_{PI} never leaves the host device and hence an adversary cannot predict the private key. However, unlike other key distribution schemes, which generate a single key-pair, our proposed scheme generates a pool of key-pairs K_{POOL}.

This pool of key-pairs K_{POOL} decreases the probability of finding the actual key, which is randomly selected from the pool by the server, amongst the rest of the dummy keys. Each key pair K_P generated uses a different version of the seed value S_V to avoid similarity in the generated pairs and will provide the same level of encryption. All key-pairs K_P are regarded the same until the server picks one (the chosen key pair for the device) making the rest into dummy keys.

The number of keys generated, $K_{POOL,}$ is directly proportional to the number of reachable edge devices N_E. As the key pool is divided amongst the available edge devices, $D_{E,}$ for increased randomness, more key-pairs are generated if more edge devices are in range. Each edge node is given a subset of the pool of keys N_S generated to send to the server.

The number of keys also varies based on the maximum size of data that each packet can contain as well as the key length. On average, 50 key-pairs are generated for each available edge device and only the public keys are sent out in sets.

C. Key Transmission

The scheme takes advantage of the availability of multiple nodes/points of connectivity to the network. Instead of sending all the keys through one point of contact, sub-sets of the key pool (key-set) N_S are created and distributed amongst the reachable nodes D_E (Fig. 1), thus further increasing the anonymity in the key exchange process.

The key set can be transmitted in both the static network by using a routing table with predefined paths, and the dynamic network by using a greedy algorithm and dynamically selecting the next-hop nodes). The transmission's anonymity will be less for a predefined path as compared to that of a dynamic one, but both will provide adequate anonymity to increase the overall security of the scheme.

Moreover, it provides additional protection against malicious nodes as the key pool is divided amongst the available edge nodes. This prevents the malicious node from getting hold of all possible keys. In addition to multiple paths, the packets carrying the keys are encrypted and looked like any normal data packet to make it harder to trace it across the network.

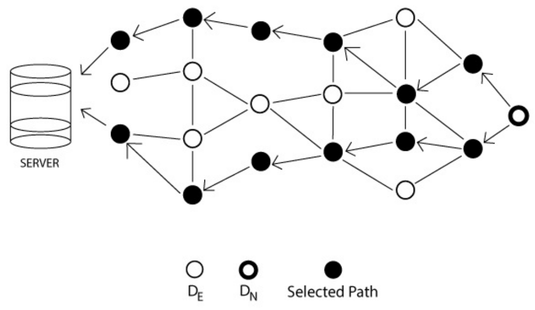


Fig. 1. Multi-Path data transmission

D. Key Selection

As the different sets of keys reach the server through different paths, it is important to know when they will expire. Hence, every set of keys sent across has a timeout value preset in it (TTL). The server will check this value when it receives the packet and will accordingly discard any packet whose TTL has expired.

The server will then randomly select a public key K_{PU} from the collective pool of all the keys received within the TTL. This key will then be registered as the communication key for the new device and can be requested by any device in the network to securely communicate with the new device.

In order to maintain two-way encryption, the server will also share a public key from its own pool using which the new device can contact the server. The server uses multiple key pairs for enabling devices on the network to communicate with it. The server, however, does not generate a unique key for each device, instead, it uses a key for a group of a predefined number of devices. Once the server's most recent public key is distributed to the pre-set number of devices, it creates a new key pair and will start allocating it to another set of new devices.

E. Challenge-Response

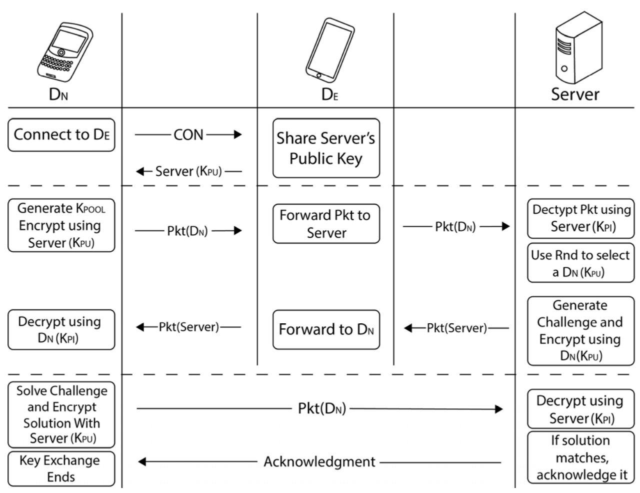
The server, upon having selected a public key, will send the new node a mathematical challenge. This is achieved by creating a moderately complex mathematical question which is then encrypted (along with the server's own public key) using the new device's selected public key. By doing so, the server can assure the new device knows which public key has been chosen.

The new device upon receiving the challenge decrypts it with its pool of private keys. Once the new device can decrypt the challenge it knows which public key has been selected by the server and linked to the new device. It will then solve the challenge and encrypt the answer using the server provided public key. This is then sent over to the server for confirmation that the new device knows the selected key and is now ready to become a part of the network.

V. Key Distribution Process

This section discusses the methodology involved in the key distribution process. The process has been divided into various steps outlined by the MPARK distribution scheme; see Table II for details regarding the notations used in this section.

Once D_N is authenticated, it will establish an open line connection to any D_E in range. The connection at present is


DN
DE
Server
Connect to DE
CON
Server (KPU)
Share Server's Public Key
Generate KPOOL Encrypt using Server (KPU)
Pkt(DN)
Forward Pkt to Server
Pkt(DN)
Dectypt Pkt using Server(KPI)
Use Rnd to select a DN(KPU)
Decrypt using DN(KPI)
Pkt(Server)
Forward to DN
Pkt(Server)
Generate Challenge and Encrypt using DN(KPU)
Solve Challenge and Encrypt Solution With Server (KPU)
Pkt(DN)
Decrypt using Server(KPI)
Key Exchange Ends
Acknowledgment
If solution matches, acknowledge it
Fig. 2. MPARK distribution at a glance

defined as an open line because no keys have been exchanged to enable encryption.

$$D_N\,(CON) \rightarrow D_E \in N_E \tag{1}$$

All the D_E will provide D_N with Server (K_{PU}). As the server maintains a set of K_{PU}'s, individual D_E may possess different K_{PU} (Server), which they use to communicate with the server.

$$\text{Server } (K_{PU}){:}\ D_E \rightarrow D_N \tag{2}$$

D_N will calculate the number of CON it has with D_E (N_E), based on which it will calculate the size of K_{POOL} to be used. The size of K_{POOL} depends upon the number of CON with D_E. A single Pkt will be sent to each D_E containing N_S of K_{PU}.

$$\sum \text{CON } (D_E) \rightarrow N_E \tag{3}$$

$$N_S = N_E \tag{4}$$

D_N will use random data for the S_V value to create a set of N_P K_P.

$$\text{Rnd } (S_V) \rightarrow D_N\ (K_{PU} \cup K_{PI}) \tag{5}$$

$$N_P{:}\ (D_N\ (K_{PU} \cup K_{PI})) = N_P{:}\ D_N\ (K_P) \tag{6}$$

$$N_P{:}\ D_N\ (K_P) = K_{POOL} \tag{7}$$

The generated K_{PU} will be divided into N_E sets and individually encrypted with the server K_{PU} provided by each D_E.

$$\textstyle\sum^{N_E} \text{Server } (K_{PU}) \rightarrow \sum^{N_S} D_N\ (K_{PU}) = N_S\ (\text{Pkt}) \tag{8}$$

The encrypted N_S will be converted into packets and will be given a TTL value. Then they will be distributed to the D_E corresponding to the server K_{PU} they provided.

$$\text{TTL} \cup N_S\ (\text{Pkt}) = \text{Pkt } (N_E) \tag{9}$$

$$\text{Pkt } (N_E) \rightarrow D_E\ (N_E) \tag{10}$$

Once D_E receives the encrypted Pkt, it will be sent across the network to the server. Each Pkt may take different time and a different path to reach the server.

$$\text{Pkt (DN): DE} \rightarrow \text{Server} \tag{11}$$

The server will collect the Pkt's with a valid TTL; any late Pkt will be discarded. The server will then decrypt the received Pkt with its K_{PI} linked to the respective K_{PU} used to encrypt the Pkt.

$$\text{Server } (K_{PI}) \rightarrow \text{Pkt } (D_N) = \textstyle\sum^{N_S} D_N\ (K_{PU}) \tag{12}$$

Once all the N_S (D_N) that made it to the server have been collected and decrypted, the server will then use a Rnd to select a single D_N (K_{PU}) from the N_S.

$$\text{Rnd}: \sum^{N_S} D_N (K_{PU}) \rightarrow D_N (K_{PU}) \tag{13}$$

Once a K_{PU} is selected, the Server will then prepare a mathematical challenge for D_N. The Server will encrypt this Mathematical challenge with the D_N (K_{PU}) it has selected. It will also include the Server (K_{PU}) for D_N to respond using a secure channel.

$$D_N (K_{PU}): (4 * 8) \cup \text{Server} (K_{PU}) \rightarrow \text{Pkt (Server)} \tag{14}$$

The server will then send the challenge over to D_N for confirmation of D_N (K_{PU}) selection and verification.

$$\text{Pkt (Server)} \rightarrow D_N \tag{15}$$

Once D_N receives the Pkt from the Server, it will start the process of decrypting it using its K_{POOL} of K_{PI}. In doing so, D_N will find out which K_P has been selected by the server.

$$\sum^{K_{POOL}} D_N (K_{PI}) \cup \text{Pkt (Server)} \rightarrow D_N (K_P) \tag{16}$$

$$D_N (K_P) \rightarrow D_N (K_{PI} \cup K_{PU}) \tag{17}$$

As D_N now knows the K_P chosen, it will use the K_{PI} to decrypt the Pkt received from the server to obtain the challenge. In addition, it will also get the K_{PU} of the server.

$$D_N (K_{PI}) \rightarrow \text{Pkt (Server)} \rightarrow (4 * 8) \cup \text{Server} (K_{PU}) \tag{18}$$

The D_N will then solve the challenge and as before, it will encrypt the answer using Server (K_{PU}) it just received. D_N will also include its K_{PU} to confirm its selection.

$$(4 * 8) = 32 \tag{19}$$

$$\text{Server} (K_{PU}) \rightarrow ((32) \cup D_N (K_{PU})) \rightarrow \text{Pkt} (D_N) \tag{20}$$

D_N will now send the Pkt to the server and finish the key exchange process from its end.

$$\text{Pkt} (D_N) \rightarrow \text{Server} \tag{21}$$

Once it receives the Pkt, the Server will use its K_{PI} linked to the K_{PU} provided to D_N to decrypt the Pkt.

$$\text{Server} (K_{PI}) \rightarrow \text{Pkt} (D_N) = (32) \cup D_N (K_{PU}) \tag{22}$$

The Server will check the answer received and, in addition, if the K_{PU} received is the same one that was provided to D_N then the Server will finish the key exchange process.

$$(32) = (4 * 8) \rightarrow \text{True} \tag{23}$$

$$\text{Server} (D_N (K_{PU})) = D_N (D_N (K_{PU})) \rightarrow \text{True} \tag{24}$$

VI. Security Analysis

In this section, we discuss the claims brought forward by the proposed MPARK distribution scheme and their proof. We will also analyze the strength of our proposed scheme against known security threats relevant to distributed networks.

A. Validation

The proposed key distribution scheme makes the following claims:

Claim 1: The proposed scheme introduces a multi-key approach making it computationally complex for the intruder to discover the communication key being used.

Proof: According to the MPARK distribution scheme, a new device must create a pool of keys amongst which one of the keys is selected by the server as the communication key for the new device. As all the keys generated have identical strength and are of the same length, it is computationally complex to determine which one would be selected as the communication key for the new device.

$$P_1 \text{ (Key Discovery)} = 1 / K_{POOL} \tag{25}$$

As we can see from the equation above, the probability P_1 of a key being found is inversely proportional to the number of keys generated as defined by the size of K_{POOL}.

Claim 2: The proposed scheme incorporates a multi-path approach to distribute keys to avoid potential internal threats.

Proof: In addition to using a pool of keys, public keys are organized into a set of keys, which are then sent across the network to the server with the help of multiple edge devices. Each edge node uses a different path to transmit the key set.

The use of these organized set of public keys ensures the server at least has a sub-set of the K_{POOL} to pick a key from. The number of keys in each set is determined by the number of edge devices in range of the new device.

$$N_S = K_{POOL} / N_E \tag{26}$$

$$P_2 \text{ (Key Discovery} \mid P_1) = P_1 / N_S \tag{27}$$

By distributing the generated keys into multiple devices, the probability P_2 of key discovery is further reduced based on the number of edge devices available. As seen in the equation above, P_2 is further reduced when P_1 is added to the equation. In addition, if the nature of the path is dynamic, we can consider additional randomness at each hop taken by each set of keys as an additional improvement over P_2.

Claim 3: By incorporating anonymity in the key exchange process the proposed scheme ensures data packet confidentiality by concealing source information.

Proof: The data packets used in the MPARK distribution scheme are disguised as normal data packets. In addition to the disguise, the source information of the packet is retracted from the header and is replaced by the last hop information. The source information is added to the encrypted part of the message. This information is only intended for the server because only the server will be able to decrypt the packet.

This ensures that while the packet travels through the network, its source and data cannot be identified or tracked providing anonymity.

Claim 4: Use of challenge-response in the proposed scheme makes it computationally improbable to tamper selected key.

Proof: According to the MPARK distribution scheme, once the server has selected a key, it does not relay it back to the new device, but instead uses a challenge-response. As the new device possesses all the private keys, the server creates a mathematical challenge, encrypts it with the selected key and sends it to the new device.

By using a challenge-response, the selected key does not need to transmit back, just the encrypted message. As only the new device can decrypt the message using its pool of private keys, it will know which public key was used by the server to encrypt the packet. It can then send the solution to the challenge provided by the server confirming the new device now knows which public key was selected and thereby ending the key exchange.

B. Threat analysis

As devices in a distributed network are not usually monitored, there are various types of attacks, some affect their vicinity whereas others may affect the whole network [12,13]. Here, we discuss major security threats in key distribution and how MPARK distribution scheme validates against them.

Rogue / Compromised Network Device: In the event of a network device being compromised or going rogue, there is the very certain risk of the network becoming unstable. The major risk is the use of symmetric keys or the same set of asymmetric keys for the entire network. The rogue/compromised device will have the capability to intercept and disrupt any exchange on the network whether it is an open line or encrypted.

In our scheme, if a device (or multiple devices) is compromised, it would not affect the network. As described in Section III.A, each device uses its unique key-pair to communicate with other devices and the server. In addition, only the public keys are ever distributed, and the private key never leaves the host device; ensuring that only the host device can decrypt the data destined for it.

(Distributed) Denial of Service Attack: A DOS or DDOS attack may not have that much impact on network devices as overcrowding a device would just result in the data taking a different and less congested path. A DOS / DDOS attack on the server can result in relative delays in the network by slowing down the requests. However, this can be overcome by the server using a detection system as it already has enough power and resources to run it. It can also be resolved if the multiple synced servers are used to balance the request loads.

Rogue Edge Device: In the scenario that the edge devices involved in the authentication process turns out to be a rogue device, the key exchange may be compromised or tampered with. The MPARK distribution scheme uses multiple edge devices to initiate the key exchange process, as described in Section III.C, to make sure that a subset of all the keys generated is successfully transferred to the server through other edge devices. In addition, the pool of keys being sent is encrypted with different server public keys making it highly improbable for the rogue edge device to decrypt it.

Acknowledgment spoofing: A compromised device can disrupt an ongoing key exchange with the server by sending a spoofed acknowledgment message to the device or even the server. This attack can be used in addition to other attacks to avoid the attack being detected by neighbor devices or the server. This is avoided using a mathematical challenge that can only be solved by the device whose public key was used to encrypt the challenge in the first place. As described in Section III.E, the MPARK distribution scheme uses a mathematical challenge to acknowledge the receipt and selection of the public key for the new node, thereby avoiding any spoofed acknowledgment.

Man In The Middle Attack (MITM): A MITM attack during the key distribution process is a major threat. It can result in tampering of the key, impersonation and even identity theft causing a major network failure. This is achieved by using a combination of randomness in the key transmission and selection process along with the anonymity achieved by disguising packets and the use of dummy keys.

When combined, the multipath approach, random key generation and selection, anonymous key transmission, mathematical challenge and the use of dummy keys significantly decrease the probability of the selected key being tampered with.

VII. Conclusion

The key distribution scheme is a vital component of the security framework of any distributed network. In this paper, we proposed a new key distribution scheme designed for distributed networks. Its major components were discussed in detail along with the methodology and then analyzed based on various threats. The proposed MPARK distribution scheme provides a unique combination of randomness, anonymity and multi-path approach which can prevent well-known threats by minimizing the probability of compromise during the key exchange process.

References

[1] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks," ACM Transactions on Sensor Networks (TOSN), 2006, pp. 500-528.

[2] S. Ruj, A. Nayak, and I. Stojmenovic, "Pairwise and triple key distribution in wireless sensor networks with applications," IEEE Transactions on Computers, 2013, pp. 2224-2237.

[3] D. Liu, P. Ning, and R. Li, "Establishing pairwise keys in distributed sensor networks," ACM Transactions on Information and System Security (TISSEC), 2005, pp. 41-77.

[4] D. K. Altop, M. A. Bingöl, A. Levi, and E. Savaş, "DKEM: Secure and efficient distributed key establishment protocol for wireless mesh networks," Ad Hoc Networks, 2017, pp. 53-68.

[5] O. K. Sahingoz, "Large scale wireless sensor networks with multi-level dynamic key management scheme," Journal of Systems Architecture, 2013, pp. 801-807.

[6] X. He, M. Niedermeier, and H. D. Meer, "Dynamic key management in wireless sensor networks: A survey," Journal of Network and Computer Applications, 2013, pp. 611-622.

[7] S. A. Camtepe, and B. Yener. "Key distribution mechanisms for wireless sensor networks: a survey," Rensselaer Polytechnic Institute, Technical Report, 2005, pp. 05-07.

[8] H. Chan, A. Perrig, and D. Song. "Random key predistribution schemes for sensor networks," IEEE Symposium on Security and Privacy, 2003.

[9] L. Eschenauer, and V. D. Gligor. "A key-management scheme for distributed sensor networks," Proceedings of the 9th ACM Conference on Computer and Communications Security. ACM, 2002.

[10] B. Wu, J. Wu, E. B. Fernandez, M. Ilyas, and S. Magliveras, "Secure and efficient key management in mobile ad hoc networks," Journal of Network and Computer Applications, 2007, pp. 937-954.

[11] M. Abolhasan, B. Hagelstein, and JC-P. Wang. "Real-world performance of current proactive multi-hop mesh protocols," 15th Asia-Pacific Conference on. IEEE Communications, APCC 2009, 2009.

[12] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges," Future Generation Computer Systems 78, 2018, pp. 680-698.

[13] C. Karlof, and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures." Ad hoc networks 1.2-3, 2003, pp. 293-315.

[14] M. S. Obaidat and N. Boudriga," Security of e-Systems and Computer Networks," Cambridge University Press, 2007.

[15] Mohammad S. Obaidat, I. Traore and I. Woungang," Biometric-based Physical and Cybersecurity Systems," Springer, 2019.