Securing Tactical Ad-hoc Networks Based on CRAN Protocol

David Lopes
Academia Militar, Lisboa
Instituto Superior Tcnico, Lisboa
Email: david.f.r.lopes@tecnico.ulisboa.pt

Abstract—**The most valuable commodity of this century is information. Its aquisition allows commanders in the battlefield to make better and faster decisions. Consequently, it is as important to deny the enemy access to information, as it is for our forces to acquire it.**

The existence of battlefield management systems allows the commander of a specific force to access useful information, aiding the process of decision making. These systems are considered by the enemy as a rewarding target. This happens due to the concentration of information regarding our forces and the battlefield. With that said, the safety of these systems must be a priority.

CRAN is a protocol that provides support to these systems and it was designed to work at the lower hierarchical levels, form platton to battalion. However, the security of this protocol was not designed, leaving its concern to other means.

In the present study, a key distribution system for mobile Ad-hoc networks is designed, which significantly increases the security of the CRAN protocol. The aim of this system is that the exchange of CRAN's messages, among participating elements, occurs under a cipher. This cipher is generated by the key that this system is designed to manage safely. This way, confidentiality and data integrity are ensured to CRAN messages. The proposed system features include adition/removal of nodes to/from the network and, consequently, key refreshing. It also has the capability to adapt in real time to the destruction of any node and to the partitioning of the network into several groups.

The impact of the proposed system was measured for different scenarios and node quantity, focusing primarily in time needed for the network to become coherent with reality. Network traffic was also evaluated along with the amount of lost and, consequently, resent messages.

1. Introduction

Nowadays, in the battlefield, it is extremly important to make the best decisions possible, and to do so in the fastest way, it is necessary to have every useful information available. This concerns to the higher hierarchical levels aswell as to the lower ones. This information is acquired by the forces present in the battlefield and it is pertinent to make it reach every element within the unit. With that said, there is an effort made by the portuguese armed forces, in the development and improvement of battlefield management systems (BMS). These BMSs aim to provide continuous updates of the battlefield state, increasing our forces awareness, which leads to a better decision making, and consequently, an improvement on the mission results.

The acquisition of information, it's denial to the enemy and the dissemination of counterintelligence, are of extreme importance to any military force nowadays, which makes BMSs rewarding targets to the enemy. In that sense, the safety of this systems must become a priority, ensuring confidentiality, data integrity, authentication, access control, protection against replay and non-repudiation.

In [1], it is proposed a protocol (CRAN) designed for mobile Ad-hoc networks (MANET), which supports BMSs, operating at the lower hierarchical levels (platoon to battalion). This protocol ensures the continuous update of it's network topology and the forwarding of tactical messages (TM). A TM is basically a message that comes from a higher layer, it is treated, by this protocol, as data to be forwarded to every node in the network. CRAN has no other concerns regarding the data within TMs.

This protocol revealed efficiency in respect to information sharing within the hierarchical levels to which it was designed to operate. All, with a high message delivery ratio and use of low bandwidth. However, it lacks in security against enemy hacks. The need to develop a key distribuition system which allows the safe use of this protocol, arrises in this context, and is adressed in this paper. Besides the protection against enemy hacks, it is necessary to consider other problems that arrise from this type of networks, such as low bandwidth availability, high latency and high mobility. This takes the problem to a trade-off between security and network funcionality.

The system proposed in this paper, aims to provide security to CRAN protocol through key distribution and its use to cipher CRAN's messages. Therefore, the system must provide the following features:

- Node adition to the network, providing the key that allows access to CRAN's messages;
- Node removal from the network, granting confidentiality of future CRAN messages;

- Node eviction from the network, in case it is needed to force a node out of the network;
- Coexistence of 2 or more units using this protocol, without mutual interference.

The system should also have the capacity to adapt in real time to network changes, namely:

- Node destruction or neutralization;
- Network partition in several groups, whether by nodes being out of reach, obstacles in the field or due to jamming.

The remainder of the paper is organized as follows. Section 2 depicts the background, in which there are adressed relevant concepts and security systems to the development of the system proposed. Section 3 presents a proposal for a key distribution system that grants security to CRAN. In section 4, shows the simulation results of the previously proposed system. Finally, section 5 describes the main conclusions of this paper and future work proposals.

2. Background

This section presents related work and concepts, which were useful to the development of the system proposed in section 3.

2.1. CRAN (*Cognitive Routable Ad-hoc Network*)

To adress the needs of communication between lower hierarchical level's unit elements, and in that way improve the results of the mission, it was proposed in [1], a Cognitive Routeable Ad-hoc Network protocol (CRAN). Figure 1 depicts the envisioned network architecture.

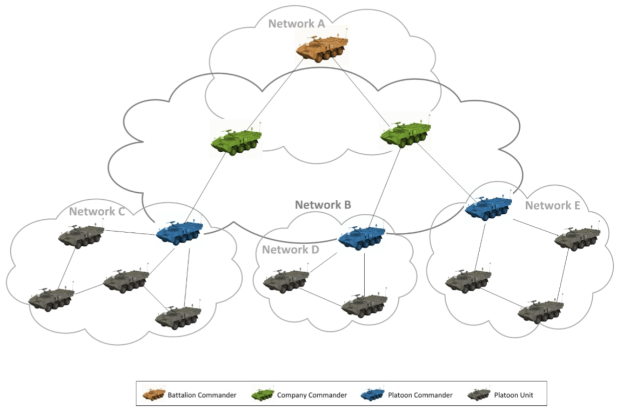

Figure 1. Network architecture [1].

This protocol, besides having in consideration the hierarchical military structure, it also takes advantage of the broadcast nature of the wireless communications to perform the node discovery and topology adaptation. CRAN contains 2 planes: the control plane, which is in charge of the node discovery and topology adaptation, and the data plane, which is in charge of forwarding TMs whithin the platoon.

The discovery process is the mechanism in which an element becomes aware of its neighbours. To do so, each element periodically sends a Keep Alive message. In case the sending element is not yet known by it's neighbours and *vice-versa*, the discovery process is initialized, where every participant elements share their network knowledge through Link Request and Link Reply messages. Basically, the discovery process works as a 3-way handshake as depicted in Figure 2 , it makes use of the 3 types of messages referred previously to ensure synchronization between participating elements.

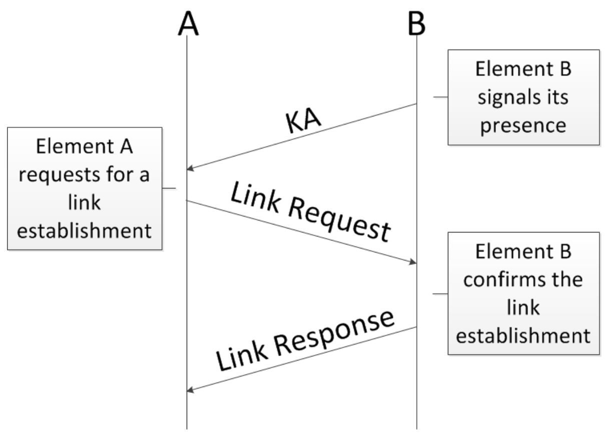
A
B
Element B signals its presence
KA
Element A requests for a link establishment
Link Request
Element B confirms the link establishment
Link Response

Figure 2. 3-way handshake [1].

Associated to this 3-way handshake, there's a finite state machine (FSM), presented in Figure 3, which ensures the delivery of the messages exchagend during the discovery process.

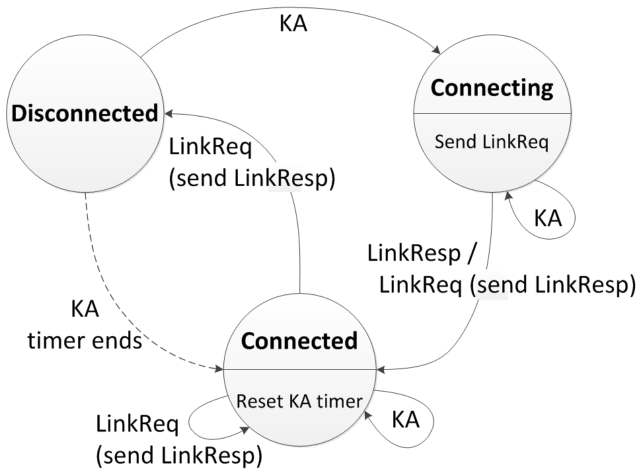

Figure 3. Discovery process FSM [1].

The topology adaptation process is in charge of the continuous update of the network topology and it's crucial in MANETs to ensure that it remains consistent. This is because there is frequent connection disruption or establishment due to changes in elements postioning. This process is initialized everytime a connection is established or disrupted. The node that detected this event, becomes responsible to forward it's information through an Update message.

The data plane, ensures the distribution of TMs throughout the platoon. Once both planes are independent, they can work simultaneously. Whenever a node needs to send a TM, it broadcasts that message only once. It's neighbours will rebroadcast it in turn, and so on. This method grants the forwarding of the TM through every platoom node.

Throughout this section, the diferent types of messages within this protocol were partially addressed, those being:

- Keep Alive - Used to announce the respective nodes presence;
- Link Request and Link Response - Used to mediate and finish the discovery process, respectively;
- Update - Used to forward the awareness of topology changes;
- Tactical Messages - Used for data sharing between every network elements.

The formats of this message types are presented in figure 4.

MsgType	PlatoonKey	SourceId

i. KA frame

MsgType	PlatoonKey	SourceId	DstId	<ElementsInfo>

ii. Link Request/Response frame

ElementId	PlatoonKey	Forwarder	Metric

iii. ElementsInfo item format

MsgType	PlatoonKey	SourceId	<ElementsInfo>

iv. Update frame format

MsgType	MsgNumber	SourceId	FragNumber	TotalFrag	<TacticalInfo>

v. TM frame format

Figure 4. CRAN messages format [1].

The results presented by this protocol, concerning the maintenance of network topology data in each node, show that the bandwidth required is reduced and matches the one available in military scenarios. Regarding the elapsed time between detecting a change in network topology and disseminating it to every node in the network, is in the order of hundreds of miliseconds. It also depends on the topology of the network at that time and increases proportionally with the number of nodes.

On a side note, even though the results have been obtained through tests executed in military scenarios, this protocol may also be used in civil scenarios.

2.2. Encryption algorithms

2.2.1. Symmetric-key ciphers. According to [2], symmetric-key ciphers are based on the use of two keys. Those keys are known by the nodes that intend to comunicate, one by each node. Each of the keys can decrypt a message originally encrypted by the other. However, in most practical symmetric-key schemes, both keys are the same, allowing the existence of only one key, which can decrypt a message previously encrypted by that same key. This also allows the comunication between more than 2 nodes.

It is extremly important that no other node, that is not in the communication group, has access to the key. Otherwise, the confidentiality of the communication is compromised. That's why there are issues in finding efficient methods to exchange keys securely, if they were not previously exchanged [2]. Those problems are called, key distribution problems and will be introduced further in this paper.

Onde more, according to [2], the most well known symmetric encryption algorithms are, Data Encryption standard (DES) and Advanced Encryption Standard (AES).

2.2.2. Public-key ciphers. According to [3], public-key ciphers are based on the use of 2 different keys: public key and private key. The public key may be known by every node in the network, meanwhile the private key must be known only by the node to which the encrypted message is sent to. None of this keys can decrypt a message that was encrypted by itseld in the first place. This way, it is possible to spread the knowledge of the public key throughout the network, once only the node that knows the private key is able to decrypt the messages encrypted by the public key. For that reason, the private key can't be known by any other node, in order to avoid the decryption of messages by them, compromising the confidentiality of the network. It is also importante to state, for the reasons mentioned above, that it should not be possible to obtain one of the keys from the other.

The use of this cipher schemes, carry a higher computational cost comparing to symmetric-key's. The most used schemes are, Rivest Shamir Adleman (RSA) and Elliptic Curve Cryptography (ECC). The use of this ciphers, allows a safe key distribution between nodes and digital signing of messages, something that is not possible with symmetric-key schemes.

According to [4], for the same security level, Elliptic Curve algorithms, make use of smaller parameters when compared to RSA. This smaller parameters, translate into advantages regarding cipher's processing speed, key sizes and, consequently, certificates. This feature was extremely important for the system's development, presented in section 3.

2.3. Digital signature

As mentioned in [2], digital signature is a primordial cryptographic concept to ensure authentication, data integrity and non-repudiation of a message. This concept provides a mean to bind an entity to a piece of information, i.e. a message.

The signing process is done by, transforming the message and some secret information (usually, a private key) into a tag. This is called signature, and it denies the possibility for other entities, to falsificate it, once they do not have acess to the private-key. Finally, the pair (message, signature) is sent, allowing any other entity to receive the message with garanties of authentication, data integrity and non-repudiation. To do so, the receivers of the message must verify the signature using the sender's entity public key.

The use of this primitive is essential to the proposed system in section 3, once this is what ensures the security concepts referred previously (authentication, data integrity and non-repudiation).

2.4. Certificates

According to [5], certificates are strutured documents predefined in a certain way, which contain information regarding a specific entity. This certificates are issued by a Certification Authority (CA). They have a limited validity, and it can be controled in 2 ways: an expiration date contained whitin the certificate itself, or through the issuing of a revocation certificate. The last one targets a specific public key, belonging to a certain entity, stipulating that it is no longer valid, starting in a particular date.

This certificates are important to the system presented in (ref), however, the standard format X.509 usually used in the vast majority of the systems, carries a high overhead if they are to be sent through the network. Considering the implications of the system's design regarding bandwidth, the need to use smaller certificates arrises.

[6] presents several certificate models smaller than the format X.509. Even though, none of them addressed the possibility of containing optional information, it was possible to use their core ideas and implement them in the proposed system. Those ideas are based in certificate chain, in which, the higher level belongs to the CA's certificate. This certificate may contain information that is common to every other entity in the network (i.e. digital signature algorithm), allowing lower level certificates to not carry that information, therefore becoming smaller. It is important to state that, this idea is only advantageous if the CA's certificate is known by every node within the network.

2.5. Certification Authority

In [7], key management in systems that use public-key cryptography, is usually adressed with the help of a trusted entity, the CA, which issues certificates. To do so, it makes use of its digital signature, allowing authentication of the node, to which that issued certificate belongs to.

As mentioned in [8], this management needs the CA to be online at all times, due to the fact that it is the only entity capable of issuing certificates. With that said, the CA becomes a critical point in the network, if it is compromised or its private key is found, then a malicious entity can easly issue forged certificates or revoke previously issued ones. An aproach to this problem lies in the replication of CAs. However, it increases the system's vulnerability, because it only takes one of its replics to be compromised, leading to the same issues mentioned above.

2.6. Key management in MANETs

In MANETs, it is important to consider that a node operates as a terminal and as an intermediate router in order to forward traffic to other nodes. Hereupon, the existence of a malicious node acting as an intermidiate one, which its objective is to intercept communications, needs to be considered. Besides the security features that that group key management should ensure, in MANETs, other factors need to be considered, like node mobility, available bandwidth and battery consumption.

In order to adress this concerns, [9] proposed a key distribution system in which its core idea lies on the existence of a group leader. This leader is responsible for the key management within the group. This idea is interesting in the resolution of the problem proposed in this paper, since the group leader might be the node of the unit commander, taking charge of the afore mentioned responsibilities.

This system defines 4 types of keys, however, only 2 of these types are imporant for this matter, namely:

- Group key (symmetric-key) - Used by every group member, to encrypt and decrypt messages shared within the group;
- Public and private key pair - Every node has a pair of these keys associated to it, used mainly for safe distribution of the group key.

The group leader generates a group key for its group. This key is refreshed everytime a member is added or removed from the group, ensuring perfect forward secrecy (PFS) of the messages shared within the group.

2.6.1. Member adition. The first thing to be done after a request to join the group is made by a node, is an authentication challenge between the leader and the requestor. Unfortunatly, [9] does not clarify how this challenge is made.

After the authentication challenge is completed, the requestor sends its certificate to the leader. Once received, the leader verifies if the certificate is valid and obtains the requestor's public key. Afterwards, the leader sends to the requestor a new message encrypted with its public key, containing: the assigned identification and a shared key (symmetric key), shared between them both. The following step includes the update of the group member's list and a the generation of a new group key. This data is sent to the new group member, encrypted by the shared key.

Finally, the leader sends the new group key aswell as the updated group member's list, encrypted by the former group key, to every other group member. Figure 5, depicts the described process.

2.6.2. Member removal. In order to remove a member from the group, the leader generates a new group key and updates the group member's list without the node targeted for removal. Afterwards, the leader sends the new group key and list data to every member (excluding the removed one), encrypted by the public key of each respective member. This process is presented in Figure 6.

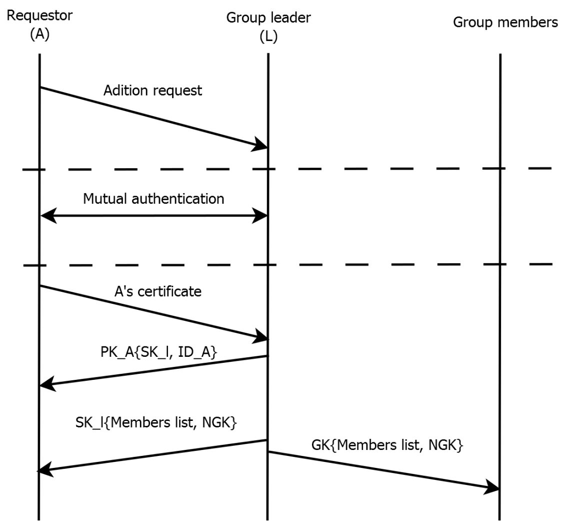

Figure 5. Member adition process.

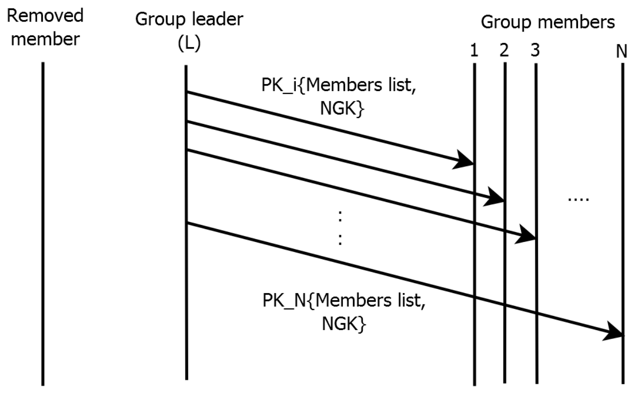

Figure 6. Member removal process.

3. Implementation

The system proposed in this paper was designed taking into account the technical features of a MANET with low bandwidth. This system aims to ensure confidentiality, integrity, authentication, protection against replication and non-repudiation of CRAN's messages. The primordial idea is, therefore, to use the minimum possible bandwidth without compromising security. To this end, 5 different message types were created: Join Request, Join Reply, Leave Request, Leave Reply and Key Refresh. Nevertheless, some fields were added to an already existing CRAN message, the Keep Alive message.

It was also taken into account the type of keys needed to ensure the functioning of the system.. Other concepts, such as identifiers, patents and certificates will be presented throughout this section, as well as their contribution to the functioning of the security system

The way this system was designed, took into account the hierarchical structure of the Armed Forces (AF), therefore, the existence of a leader is crucial. The process of leader election was based in a previously designated hierarchy, similar to real election process in the AFs, where, the coomander (leader) the highest ranked soldier. This process has the capability to elect a new leader on-the-fly if, for some reason, the leader node is out of reach or neutralized. With this, there is always an entity responsible for the group. This process may also occur when there is a group partition into several subgroups (whether it happens due to nodes being out of reach, or obstacles that prevent communications to reach their destination), making sure that there is always a leader within each subgroup.

The leader node, whoever it might be, will play a key role in the system's functioning, since it is the one with the power to accept or deny requests from other nodes to join or leave the group. It also has the resposibility to refresh the group key whenever the number of elements in the group changes. Some of this system details are inline with the ones described in 2.6.

CRAN's original messages, may be shared within the group between its members under the safety provided by a symmetric-key cipher. The key used for this cipher, is the group key.

3.1. Basic concepts

3.1.1. Identifier. Each node participating in this system has an identifier (ID), assigned by the CA when the certificate is issued. This ID is unique in the whole network.

3.1.2. Rank. Similar to the IDs, each node has a rank, assigned by the CA. This concept is important to provide a hierarchical struture between the nodes, allowing the nomination of a leader throughout the network operation.

3.1.3. Certificate. According to the idea presented in subsection 2.4, the certificate model will consist in a two level certificate chain. The higher level belongs to CA's certificate. The lower level belongs to the network node's certificates, called subaltern certificates from now on.

As described previosuly, in subsection 2.4, the CA's certificate must contain every needed information according to X.509 format. Therefore, the subaltern certificates may contain less information, since some of that information is already detailed in the CA's certificate and it is the same to every node in the network. This certificate model was proposed because there is the need to send certificates through the network. To that end, they must be as smaller as possible to reduce the required bandwidth.

3.1.4. Certification Authority. In the development of this system, it was decided that the CA must be offline, the reason behind it lies in the fact of it being a rewarding target to the adversary. If it is intended for the system to be able to adapt to node neutralization, the CA cannot be online, otherwise it would be easy for the adversary to compromise the system by capturing or neutralizing it.

The CA is the entity in charge of issuing certificates to different nodes that participate in the network. Once it is offline, the certificates must be issued before the start of the network. Even though, a node has its certificate issued by the CA, only the group leader has power to accept it, or not, within its group.

3.1.5. Message type. The message types created for this system need to be discerned. With that said, and in order to maintain the structure of CRAN's messages, it is necessary to create a new field (Msg Type). This is the first field in every message type presented from now on. It is important because every message has a different algorithm to process it.

This field differentiates 7 message types, namely: original CRAN messages, KA, Join Request, Join Reply, Leave Request, Leve Reply and Key Refresh.

3.1.6. Nonce. Nonce is the designation of a field that is present in every message type designed for this system. It's principal function is to help in protection against replay. This field's generation is based on a time index. This allows the nodes to record this field at the arrival of the message and check if it was already received. If that is the case, then the message is discarded.

3.1.7. Keys. The diferrent types of keys, designed to ensure security in this system are:

- Group key - Symmetric-key, shared between group members and used to cipher every CRAN's message;
- Public and private keys (PubK and PrivK)- Each node as an assigned pair o assymetric keys, used for adition or removal of nodes within the group.

3.1.8. Potential leaders queue. Each node has this queue, it contains the ID's of the nodes that presented themselves as group leaders. Those IDs are sorted by rank. Only IDs, with higher rank then the current leader, are put in the queue. Those IDs are collected whenever a KA message is received. This queue helps the leader election process described further in this section.

3.2. Keep Alive

This is an original CRAN message type, used to announce a node presence. However, during the development of this system, 3 fields were added to this message in order to take advantage of its periodicity, so it could act as a beacon for the group leader. The added fields are:

- Timestamp (TS) - This field contains a time index, which indicates the moment that the group leader generated his KA;
- Signature - TS field is signed by the leader, therefore, this field contains that signature;
- Leader certificate - Certificate of the group leader, to which the node that generated the message belongs.

Figure 7, ilustrates the structure of this message after the added fields.

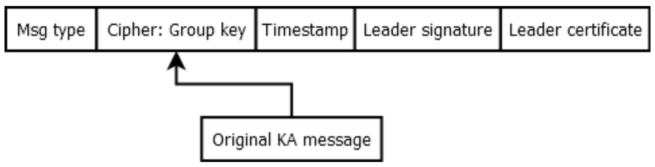

Figure 7. Keep Alive format.

The reason behind the TS field is that, the leader needs to anounce its presence within the group, at a specific time. Otherwise, it would not be possible to detect group partitions, rejoining of partitioned groups or even elect new leaders if necessary.

There are 2 reasons to why the leader's certificate must be sent in this message. First, the nodes need to have access to the leader's public key in order to validate the TS. The second, is to give an external node, knowledge of the leader's ID, in case it needs to make a request to join the group.

3.3. Node adition

This process aims to provide tools for an external node to request access to a group. It is divided into 2 message types: Join Request and Join Reply. Join Request is sent by an external node and forwarded through the group members until it reaches the leader. When the group leader receives the request, it replies by sending a Join Reply message, granting or denying access to the group. In case access is granted, the key is also sent in the Join Reply message.

Join Request message is structured by the following fields:

- Msg type - Type of message, in this case, Join Request;
- New leader ID - Identifier of the leader to which the request is sent;
- Nonce - Protects against the reuse of this message;
- External node signature - External node signs this message, therefore, this field contains that signature;
- External node certificate - Certificate of the external node, needed to verify integrity of the message and authenticate the node issuing the request.

Figure 8, ilustrates the structure of this message.

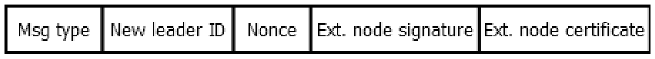

Figure 8. Join Request format.

Join Reply message is structured by the following fields:

- Msg type - Type of message, in this case, Join Reply;
- External node ID - Identifier of the node to which the reply is sent;
- Nonce - Copy of the nonce sent previously in Join Request message;
- Group key - In case the leader accepts the request, this field is filled with the group key;

- Response - Sucess, if the request was accepted. Failure otherwise;
- Leader signature - The leader signs this message.

Figure 9, ilustrates the structure of this message.

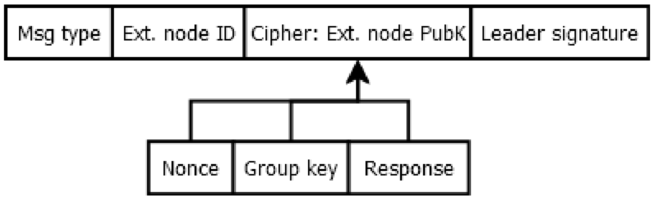

Figure 9. Join Reply format.

The fields, Nonce, Group key and Response must be encrypted using the external node public key, ensuring that only this node has acess to those fields.

3.4. Node removal

This process works similarly to node adition, however, it is used by nodes within the group to request permition to leave it. This feature is important, not only because it notifies the leader about the group changes, but also about the need of refreshing the group key. As in node adition, this process is also divided into 2 message types: Leave Request and Leave Reply. Both of this messages work similarly to Join request and Join Reply, respectively.

Leave Request message is structured by the following fields:

- Msg type - Type of message, in this case, Leave Request;
- Leader ID - Identifier of the leader to which the request is sent;
- Nonce - Protects against the reuse of this message;
- Leaving node signature - Leaving node signs this message;
- Leaving node certificate - Certificate of the leaving node, needed to verify integrity of the message and authenticate the node issuing the request.

Figure 10, depicts the structure of this message.

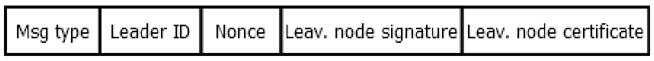

Figure 10. Leave Request format.

Leave Reply message is structured by the following fields:

- Msg type - Type of message, in this case, Leave Reply;
- Leaving node ID - Identifier of the node to which the reply is sent;
- Nonce - Protects against the reuse of this message;
- Response - Sucess, if the request was accepted. Failure otherwise;
- Leader signature - The leader signs this message.

This message's format presents a few differences regarding Join Reply's format, in this process there is no need to send the group key. Figure 11, depicts the structure of this message.

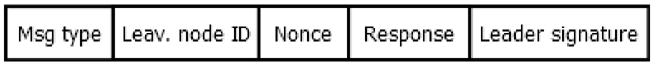

Figure 11. Leave Reply format.

After a node leaves its group, the group key is refreshed, ensuring that the left node cannot access further CRAN messages.

3.5. Node eviction

In order to remove or expell a node from the group, it is required that the leader refreshes the group key and share it among the remaining nodes. To do so, it is proposed another message type, Key Refresh (KR).

Key Refresh message is structured by the following fields:

- Msg type - Type of message, in this case, Key Refresh;
- Dest ID - Identifier of the node to which this message is sent to, being one of the remaing nodes in the group;
- Group key - The new group key goes in this field;
- Nonce - Used to avoid message replay.
- Leader signature - The leader signs this message.

Figure 12, shows the structure of this message.

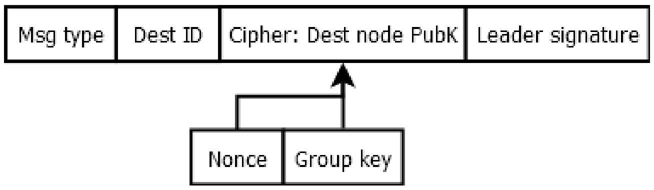

Figure 12. Key Refresh format.

The fields, Nonce and Group key must be encrypted using the public key of the node to which this message is sent to, ensuring that only this node has access to those fields. The number of sent messages is determined by the number of nodes remaining in the group. It is created a message of this type for each remaining node.

On a side note, the group leader may be out of reach. To make sure that the requests and respective replies reach their destination, the group nodes forward these messages through the group, acting as intermediates. Therefore, named intermediate nodes in this situations. This extends to the message types described in subsections 3.3, 3.4 and this one.

3.6. Other key aspects

3.6.1. Message forwarding mechanism. In order to avoid messages to be forwarded indefinetly, each intermediate node only forwards the same message once. To do so, each node saves every received message and, when the next one is received, it compares the message with the previous ones. By doing so, it is possible for a node to know if a message has already been forwarded or not. If the message was not forwarded yet, then the node does that, if not, the message is discarded.

It is also important to state that every intermediate node verifies the message integrity before forwarding it. That way, if the verification fails, the forwarding does not occur, avoiding unnecessary traffic in the network. This also prevents an adversary, from disturbing the networks functioning through the continuous injection of messages.

3.6.2. Group partitioning, leader election and rejoin. These are 3 different concepts, however they are intertwined, working together to provide adaptability to the proposed system.

The process of detecting a group partition is simple. A group partition occurs whenever the TS of the leader expires without receiving a new one. The nodes that detect this events, now know that a group partition occurred.

Afterwards, the leader election process takes place. Making use of the potential leaders queue, the node starts a node adition process. The first ID in the queue, represents the node to which the request must be done. If there is no IDs in the queue, then the node assumes leadership of the subgroup.

Assuming both subgroups become in range of each other, then they automatically rejoin. This happens due to leader election process being continuous throughout the network's lifespan. Basically, one of the subgroup leaders has higher rank than the other, therefore, the nodes from the subgroup with lower rank leader, send requests to join the other subgroup. This will happen until both subgroup merge, becoming only one.

3.6.3. PFS mode. This mode is designed to grant PFS (hence the name) to the network. To do so, everytime there is a node adition, the group key must be refreshed. This is accomplished with the use of the process described in subsection 3.5.

Due to key redistribution, when this mode is active, the network traffic increases. With that said, this mode may be activated before or during the network funtioning.

4. Performance results

This section presents simulation results obtained using NS-3 [10]. For each scenario, there were performed simulations with different numbers of subaltern nodes: 1, 5, 10 ,20 and 30 nodes. For every combination of scenario and number of nodes, 10 simulations were made. The principal point, was to obtain an average value of the time needed for the system to become coherent with the reality, and its respective 95% confidence interval.

For a better understanding of the results presented bellow, it is important to know the following:

- The period of KA messages was set to 5 seconds;
- The moments in which each node sends its KA, were distributed evenly throughout this same period (5 seconds), being that, the order defined is ascending related to node IDs;
- The leader, has always the lowest ID, therefore it is always the first one to send KAs;
- Once the simulator works synchronously, there was inserted a random delay in the sending of messages. Otherwise, the probability of colision was higher when compared to reality.

4.1. Initial group formation

In the present scenario, it is intended to evaluate the period of time necessary for a leader to be elected and all other nodes have been accepted within its group. This simulations were made with PFS mode, both active and inactive. In this scenario, every node is in range of each other, therefore it's spacial distribution as no effect in the end results.

The results obtained are depicted in Figure 13. As it is observable, when PFS mode is active it takes longer for all the nodes to join the group. This becomes more evident as the number of nodes increases.

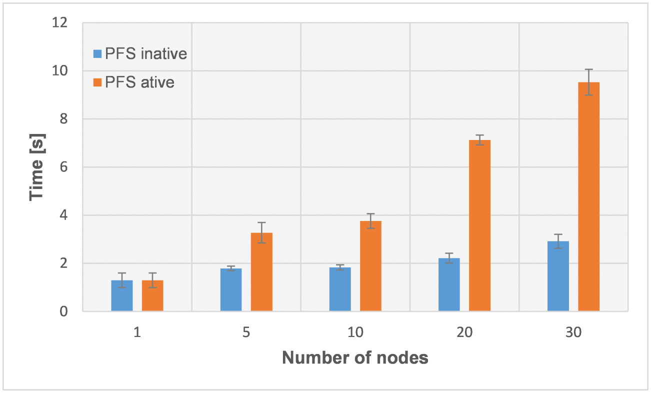

Figure 13. Average time and confidence interval.

4.2. Initial group formation (3 *hops*)

The only diference between this scenario and the previous one, lies only in the spacial distribution of the nodes, in which it was intended, for some nodes, to be at a maximum of 3 hops from the leader. That distribution is ilustrated in Figure 14, in which the red node represents the group leader and the lines represent the nodes in range of each other.

The times obtained in this scenario are significantly higher in comparison to the previous one. This is caused

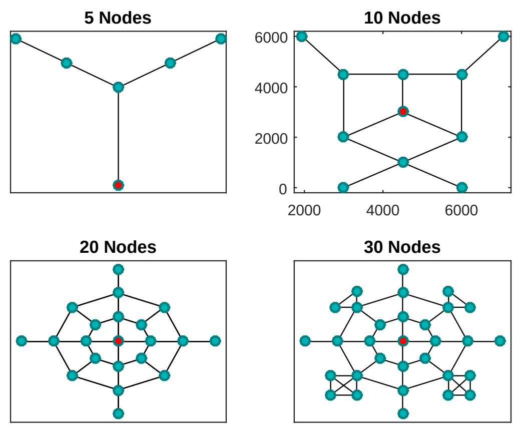

Figure 14. Spacial node distribution.

by 2 main reasons. One of them, is the increased network traffic, since it is required an aditional effort from intermidiate nodes in the forwarding of request to join the group. The other one, is simply due to the fact that a node that is 3 hops away from the leader, only receives its information after the nearer nodes being in the group. In other words, the nodes in range of the leader join the group in the first place, then the nodes in the outer layers can join the group, layer by layer. Therefore, nodes in the third layer take longer to join the group. Figure 15, corroborates the above statements.

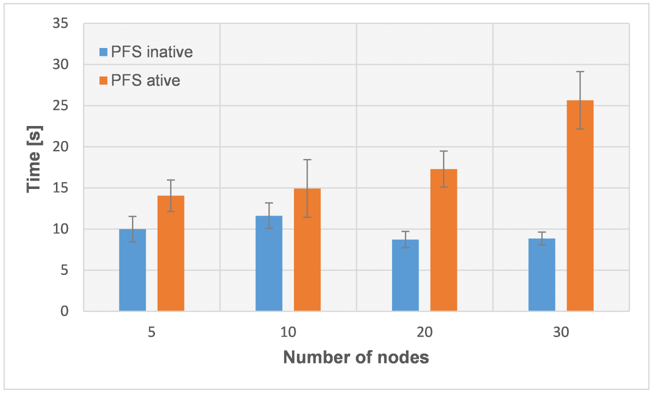

Figure 15. Average time and confidence interval.

4.3. Node leaves group

In this scenario, it was intended measure the time elapsed when one node leaves the group. The elapsed time is measured since the node sends a request to leave the group until it is accepted and the group key gets refreshed. In this scenario, every node is in range of each other. Also, PFS mode does not make a diference in this scenario. Simulation results are shown in Figure 16.

In the present scenario, the process of refreshing the group key is the most demanding, in what time is concerned. Therefore, as the number of nodes within the group increases, also increases the amount of KR messages being sent. With that said, the probability of colision between messages, and consequently their loss, rises. With lost messages, resends need to be done, taking longer for the new group key to be known by every remaining node, each time the group size increases.

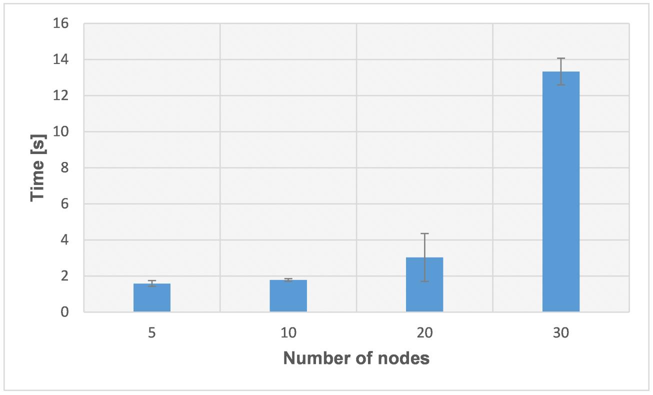

Figure 16. Average time and confidence interval.

Once, there is no key refreshing process for a 1 node simulation, it was not considered in the results shown.

4.4. Group merge (rejoin)

For this last scenario, it was intended to measure the time taken to rejoin 2 previously partitioned subgroups, into the original group. This scenario is quite similar to the previous one, where the majority of the time is taken by the key refreshing process.

The results for this simulations only matter in case the group keys of both subgroups, are different. Otherwise, the merge would only be measured by the time needed for every nodes to receive the highest ranked node's KA. If both keys are different from each other, the need for node adition processes arises.

Figure 17, shows the average time needed for every node to be within the same group and its respective confidence interval.

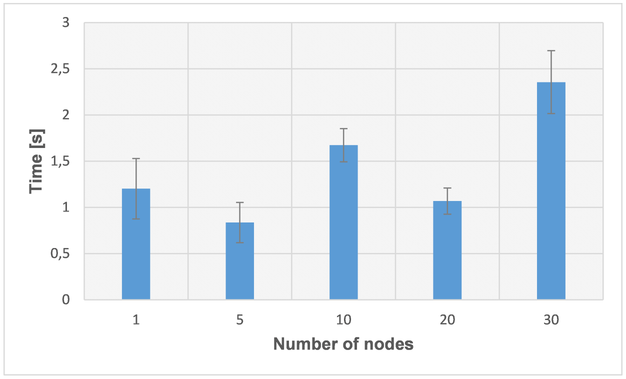

Figure 17. Average time and confidence interval.

5. Conclusion

This section presents the conclusions drawn from the development and simulation of this system. It is divided into two parts, final considerations regarding the study developed and proposals for future work.

5.1. Final considerations

The system proposed in this paper, aims primarily, to deny information about our forces to the adversary. It also intends to deny the disruption of BMSs through attacks to CRAN protocol, whether these might be by denial of service or injection of fake information.

The purpose of these system is to grant security to CRAN protocol, through a key distribuition within the network and the use of that key (group key) to cipher CRAN's messages. Only a node within a group has access to that group's key, therefore, some features were designed in order to provide access control, namely: node adition, removal and eviction. This control is enforced by he group leader, which is elected dynamically, using ranks has criteria.

Some other challenges to the development of this system were, the need for the network to adapt to partitions and node destruction. That was accomplished by the conjunction of the detect partition, leader election and rejoin processes.

Even though, the system was developed specifically to attend CRAN's features, its functioning is independent. Therefore, the designed system might be used to ensure security to other MANET protocols with similar behaviour.

Throughout the simulations, it was possible to apply the defined concepts and test the accomplishment of its objectives. The increasing number of nodes for each simulated scenario, allowed the understanding of its impact in system's performance, with or without PFS mode active.

5.2. Future work

The possibility to perform tests in a real scenarios and evaluate its impact in communications and coherence of the Common Operational Picture (COP), would be an important asset to this system.

In order to improve the results obtained, a future work might consider making use of CRAN's neighbour and topology table, to reduce the need for every node within the group to forward messages. Only the nodes that provide the shortest route to any other, would need to forward these messages.

The study of the time needed to encrypt, decrypt, sign and verify digital signatures, would also be important to take into account for future works in this system. Hence, the proposed work should focus primarily on the impact of these times, in the network's functioning.

Acknowledgments

The author would like to thank:

- His family for the unconditional support given throughout his life.
- His supervisors for their guidance during the whole study;
- CRITICAL Software's staff involved in the development of CRAN protocol;
- His friends and comrades, for their ever present support.

References

[1] G. Gomes, A. Zquete, and S. Sargento, "Self-adapted protocol for intra and inter-echelons communications," in *International Telecommunications Network Strategy and Planning Symposium - networks*, vol. 1, June 2014, pp. 1 – 1.

[2] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, *Handbook of Applied Cryptography*, 3rd ed. Boca Raton, FL, USA: CRC Press, Inc., 2006.

[3] N. C. Fernandes, "Anlise de ataques e mecanismos de segurana em redes ad hoc," Master's thesis, Universidade Federal do Rio de Janeiro, Escola Politcnica, Dezembro 2006.

[4] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006.

[5] A. Zúquete, "Segurança em redes informáticas," *FCA Editora*, 2006.

[6] M. Nyström and J. Brainard, "An x. 509-compatible syntax for compact certificates," in *Secure NetworkingCQRE [Secure]99*. Springer, 1999, pp. 76–93.

[7] A. L. d. S. Eduardo da Silva, "Implementao de um esquema de gerenciamento de chaves auto-organizado para redes ad hoc mveis," 2007.

[8] L. Zhou, L. Z. Department, and Z. J. Haas, "Securing ad hoc networks," 1999, cornell University.

[9] K. K. Chauhan and A. K. S. Sanger, "Securing mobile ad hoc networks:key management and routing," *CoRR*, vol. abs/1205.2432, 2012. [Online]. Available: http://arxiv.org/abs/1205.2432

[10] *ns-3 Tutorial*, ns-3 Project, Jan. 2010. [Online]. Available: http://www.nsnam.org/docs/release/tutorial.pdf