TÉCNICO LISBOA

Segurança em Redes Ad-hoc Táticas Baseadas no Protocolo CRAN

David Filipe Rodrigues Lopes

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador(es): Prof. António Manuel Raminhos Cordeiro Grilo

Prof. Carlos Nuno da Cruz Ribeiro

Júri

Presidente: Prof. Nuno Cavaco Gomes Horta

Orientador: Prof. António Manuel Raminhos Cordeiro Grilo

Vogal: Prof. André Ventura da Cruz Marnoto Zúquete

Outubro 2016

Dedicado a toda a minha família.

Agradecimentos

A presente dissertação é fruto de um trabalho individual, mas que não teria sido possível a sua conclusão sem o apoio das pessoas que, de alguma forma, contribuíram para a sua realização. Neste contexto, aproveito este ponto para expressar os meus agradecimentos às seguintes pessoas:

- À minha família, em particular aos meus pais, por me terem apoiado em todos os aspetos da minha vida e me terem proporcionado o dom da educação. Graças aos seus incentivos foi-me possível transpor as barreiras que se colocaram neste longo caminho que é a vida;
- Aos meus orientadores, professor António Grilo, professor Carlos Ribeiro e professor António Serralheiro pela sua pronta disponibilidade e orientação desde o início até ao fim desta jornada;
- A todos os membros da CRITICAL Software envolvidos no desenvolvimento do protocolo CRAN, por me terem proporcionado acesso ao material sempre que solicitado, bem como na atenciosa resolução de dúvidas ao longo deste estudo;
- A alguns dos meus camaradas, pelo incentivo e auxílio que me deram ao longo de todos estes anos de curso;
- A todos os meus amigos, que de uma forma ou de outra, me auxiliaram não só durante o desenvolvimento da presente dissertação, mas também ao longo da vida.

Resumo

O bem mais precioso deste século é a informação. É a aquisição desta que permite aos comandantes nos teatros de operações tomarem melhores decisões e de forma mais rápida. Por consequência, é tão importante impedir que o inimigo obtenha informação, como a obtenção desta por parte das nossas forças.

A existência de sistemas de gestão do campo de batalha visa fornecer ao comandante de uma força informação útil no auxílio a essa tomada de decisão. Estes sistemas são considerados um alvo remunerador por parte do inimigo, uma vez que concentram informação acerca das nossas forças e do campo de batalha. Assim sendo, a segurança destes deve ser considerada uma prioridade.

O CRAN é um protocolo de suporte a estes sistemas e foi projetado para que opere ao nível dos escalões pelotão até batalhão. Contudo, a sua segurança não foi contemplada, sendo deixada a cargo de outros meios.

No presente estudo, é desenvolvido um sistema de distribuição de chaves em redes móveis Ad-hoc que permita a utilização do protocolo CRAN de forma segura. Pretende-se com isto que as mensagens projetadas pelo protocolo sejam trocadas entre elementos participantes sob uma cifra. Cifra essa que é realizada pela chave que este sistema visa distribuir de forma segura, assegurando assim capacidade de confidencialidade e integridade às mensagens do protocolo CRAN. As características deste sistema passam pela adição/remoção de nós à/da rede e consequente atualização da chave. Tem ainda a capacidade de se adaptar em tempo real à destruição de qualquer nó e à partição da rede em vários grupos.

O impacto do sistema na rede foi avaliado para diferentes cenários e número de nós, com principal foco nos tempos necessários para que o sistema fique consistente com a realidade. Outra das métricas avaliadas consiste no tráfego de mensagens na rede, bem como na quantidade de mensagens perdidas e consequentes reenvios.

Palavras-chave: CRAN, segurança, gestão de chaves, redes Ad-hoc móveis, sistemas de gestão do campo de batalha

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.

Keywords: CRAN, security, key management, mobile Ad-hoc networks, battlefield management systems

Conteúdo

Lista de Tabelas

Lista de Figuras

Lista de Abreviaturas

AC	Autoridade certificadora
AES	*Advanced Encryption Standard*
CAD	Criptoanálise diferencial
CAL	criptoanálise linear
CA	*Certification Authority*
COP	*Common Operational Picture*
CPriv	Chave privada
CP	Chave pública
CRAN	*Cognitive Routable Ad-hoc Network*
DES	*Data Encryption Standard*
ECC	*Elliptic Curve Cryptography*
ECDSA	*Elliptic Curve Digital Signature Algorithm*
GNFS	*General Number Field Sieve*
ID	identificador
KA	*Keep Alive*
MANET	*Mobile Ad-hoc Network*
NS-3	*Network Simulator 3*
PFS	*perfect forward secrecy*
RSA	*Rivest-Shamir-Adleman*
STRIDE	*Spoofing*, *Tampering*, *Repudiation*, *Information disclosure*, *Denial of service* e *Elevation of privilege*

TDEA *Triple Data Encryption Algorithm*

TM *Tactical message*

TO Teatro de operações

TS *Timestamp*

Capítulo 1

Introdução

Este capítulo apresenta os aspetos introdutórios ao relatório, a saber: a motivação e definição do problema em estudo na dissertação, objetivos da mesma e o modo como estão estruturados os restantes capítulos deste documento.

1.1 Motivação e definição do problema

Nos teatros de operações (TO) atuais é extremamente importante tomar decisões de forma mais rápida possível e, para o fazer da melhor forma possível, é necessário ter à disposição toda a informação útil, tanto a nível dos altos como dos baixos escalões hierárquicos. Esta informação é obtida pelas unidades que se encontram no TO e é imperativo fazê-la chegar a todos os elementos destas. Deste modo, existe um grande esforço realizado, por parte das forças armadas, na criação e melhoramento de sistemas de gestão do campo de batalha em tempo real. Estes sistemas de gestão do campo de batalha em tempo real visam assegurar constantes atualizações do estado do TO, aumentando a consciencialização do que está efetivamente a ocorrer neste, para uma melhor tomada de decisão, levando a uma melhoria dos resultados da operação.

A partilha da informação deve ser feita em dois sentidos: horizontal e vertical. O sentido horizontal refere-se à disseminação da informação pelos elementos de um mesmo escalão hierárquico. O sentido vertical refere-se à partilha de informação entre diferentes escalões hierárquicos, o que leva à existência de um maior cuidado na filtragem da informação relevante quando é transmitida, evitando inundar os escalões superiores, permitindo-lhes uma rápida tomada de decisão. Do mesmo modo, é necessário que a informação que é passada a um escalão inferior seja simples e concisa, de modo a permitir uma rápida e fácil interpretação das intenções do escalão superior.

A obtenção de informação por parte das forças aliadas, negação de informação ao inimigo e a disseminação de contra-informação são de extrema importância para qualquer força militar nos dias de hoje, o que faz com que estes sistemas de gestão do campo de batalha em tempo real sejam um alvo propício a ataques por parte do inimigo. Assim sendo, a segurança destes sistemas deve ser feita prioridade, de modo a garantir a confidencialidade e integridade da informação e, por sua vez, a

segurança das unidades.

Foi proposto por [1] um protocolo de comunicações projetado para uso em redes Ad-hoc móveis que trabalhasse ao nível dos escalões pelotão até batalhão. Este protocolo encarrega-se da constante atualização da topologia da rede e da disseminação das mensagens de âmbito tático pela rede. A constante atualização da topologia é de extrema importância neste tipo de redes, uma vez que, devido à elevada mobilidade dos nós, estes podem facilmente sair do alcance dos seus vizinhos, sendo necessário reajustar as rotas de reencaminhamento por toda a rede para garantir que não existam falhas na entrega de mensagens.

O protocolo proposto por [1] revelou eficiência na distribuição e partilha de informação ao nível dos escalões hierárquicos para os quais foi projetado, com um elevado rácio de entrega de mensagens e reduzido uso de largura de banda. No entanto a sua segurança contra ataques é fraca. Neste contexto, no âmbito da presente dissertação, surge o propósito de realizar um sistema de distribuição de chaves em redes Ad-hoc móveis que permita a utilização deste protocolo de forma segura, quer em operações de âmbito militar, quer em operações de âmbito civil. Além da proteção contra ataques, é necessário ter em consideração todas as restrições que advêm deste tipo de redes, como a reduzida largura de banda, elevada latência e alta mobilidade dos nós, levando à procura de um compromisso entre o nível de segurança que é possível oferecer, reduzindo o menos possível a operacionalidade da rede.

1.2 Objetivos

Como objetivo da presente dissertação pretende-se o desenvolvimento de um sistema que confira segurança ao protocolo CRAN através da distribuição de chaves pela rede e uso destas para cifrar as comunicações.

O sistema a projetar deverá ter as seguintes características:

- Adição de nós a uma rede, fornecendo-lhes a chave que permite ter acesso às mensagens CRAN;
- A remoção de nós da rede, garantindo a confidencialidade das futuras mensagens;
- A expulsão de nós da rede, caso seja necessário forçar a saída de um dos nós;
- Coexistência de duas ou mais unidades de qualquer escalão sem que interfiram mutuamente nas suas comunicações.

O sistema deverá ainda ter capacidade de se adaptar em tempo real a eventuais alterações na rede, a saber:

- Destruição ou neutralização de qualquer nó;
- Partição dos elementos da rede em vários grupos, seja por se encontrarem fora de alcance ou devido à presença de obstáculos no terreno.

Os objetivos apresentados anteriormente deverão ser atingidos através de um compromisso entre a segurança e o tráfego de mensagens na rede, uma vez que o sistema deverá operar principalmente em ambiente militar, onde um dos constrangimentos é a reduzida largura de banda disponível.

1.3 Contribuições

O trabalho desenvolvido ao longo desta dissertação proporcionou algumas contribuições que podem vir a ser uma mais-valia para futuros estudos, tanto na área de redes, como na de segurança e/ou militar. Sendo estas:

- A especificação dos mecanismos desenvolvidos que permitem a segurança do protocolo CRAN e do sistema proposto;
- A implementação do sistema desenvolvido no simulador NS-3, permitindo a sua futura utilização;
- A avaliação do desempenho e comportamento do sistema em diferentes cenários e número de nós.

1.4 Estrutura do documento

Os restantes capítulos deste documento encontram-se estruturados da seguinte forma: no capítulo 2 são abordados conceitos e sistemas de segurança pertinentes ao desenvolvimento da dissertação; no capítulo 3 é apresentada uma proposta de um sistema de distribuição de chaves para garantir segurança ao protocolo CRAN; no capítulo 4 são realizadas simulações ao sistema proposto, em diferentes situações críticas, e apresentados os resultados das mesmas; por fim, no capítulo 5 são apresentadas as conclusões retiradas deste estudo e apresentadas propostas para trabalhos futuros neste âmbito.

Capítulo 2

Estado da Arte

Este capítulo apresenta uma explicação dos aspetos considerados relevantes ao desenvolvimento do tema proposto, sendo estes: o protocolo CRAN, algoritmos de cifra, assinatura digital, certificados, autoridade certificadora e mecanismos de gestão de chaves.

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

Para fazer face às necessidades de comunicação entre elementos de unidades militares de baixo escalão, e desse modo, melhorar os resultados de uma operação, foi proposto por [1] um protocolo cognitivo para redes Ad-hoc.

Este protocolo foi projetado para unidades de escalão pelotão até batalhão, ficando os comandantes das mesmas encarregues da comunicação vertical (entre unidade subordinada e unidade de escalão superior) e os restantes elementos da unidade ficam encarregues da comunicação horizontal (entre elementos da própria unidade). Na Figura 2.1, é possível observar a arquitetura da rede, com base na qual foi projetado este protocolo.

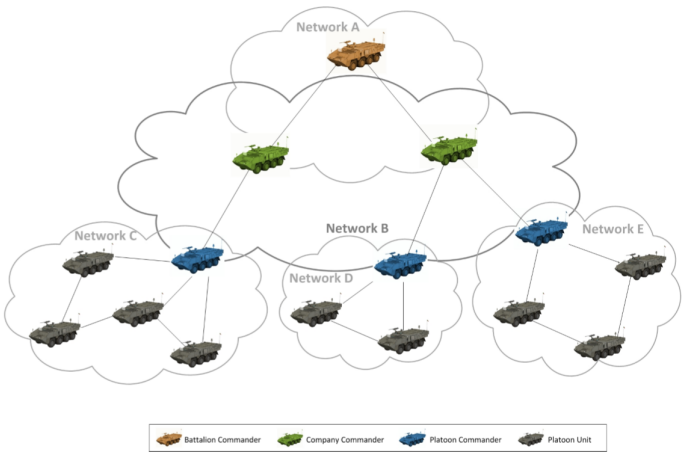

Figura 2.1: Arquitetura da rede.
[1]

O protocolo proposto por [1], além de ter em consideração a estrutura hierárquica militar , tira ainda vantagem do facto das comunicações *wireless* poderem ser recebidas por todas as unidades que se encontram ao alcance da mesma para realizar os processos de descoberta e alteração da topologia da rede, uma vez que apenas é necessário que seja enviada uma mensagem para todas as unidades ao alcance. Este protocolo está assente sobre 2 planos: o plano de controlo, que está encarregue da descoberta e alteração da topologia da rede, e o plano de dados, que está encarregue da distribuição de mensagens táticas entre o pelotão respetivo.

O processo de descoberta é o mecanismo no qual um nó toma conhecimento dos seus nós vizinhos. Para tal, cada nó envia periodicamente uma mensagem *Keep Alive* (KA), que, no caso do nó emissor ainda não ser conhecido pelos nós vizinhos e *vice-versa*, é inicializado o processo de descoberta onde os nós participantes partilham o seu conhecimento da rede através de mensagens *Link Request* e *Link Response*. Caso o nó emissor já seja conhecido pelos seus vizinhos e *vice-versa*, nada acontece, pois as suas posições, no que respeita à topologia da rede, não foram alteradas. O processo de descoberta contempla um *3-Way Handshake*, apresentado na Figura 2.2, que faz uso dos três tipos de mensagens referidos (KA, *Link Request* e *Link Response*), para assegurar a sincronização entre os nós participantes no processo.

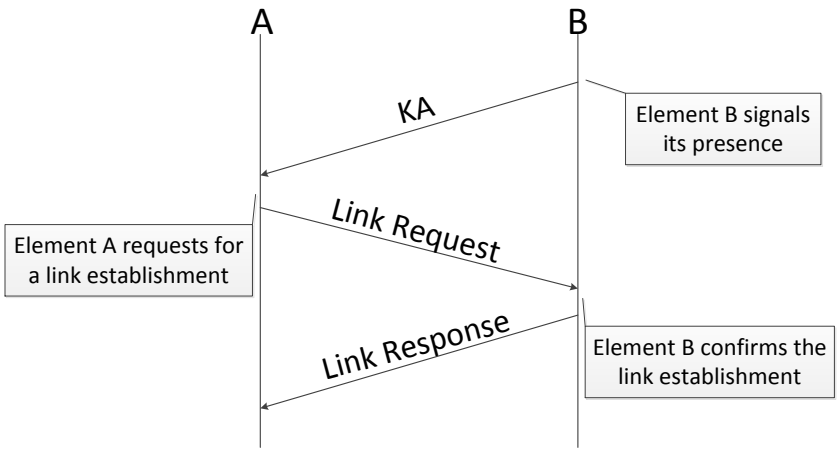

Figura 2.2: *3-Way Handshake*.
[1]

Associado a este *3-Way Handshake*, está uma máquina de estados finita (*finite state machine* - FSM), presente na Figura 2.3, que garante a entrega das mensagens trocadas durante o processo de descoberta.

O processo de alteração da topologia da rede encarrega-se da atualização desta e é essencial em redes Ad-hoc móveis para garantir que esta se mantenha coerente com a realidade, uma vez que existe, com frequência, disrupção ou estabelecimento de uma ligação devido à alteração do posicionamento dos nós. Este processo é inicializado sempre que seja detetado/a um/a estabelecimento/disrupção de uma ligação por parte de um nó presente na rede, ficando o nó que detetou esse evento encarregue de o propagar para os outros nós da rede, através de mensagens do tipo *Update*.

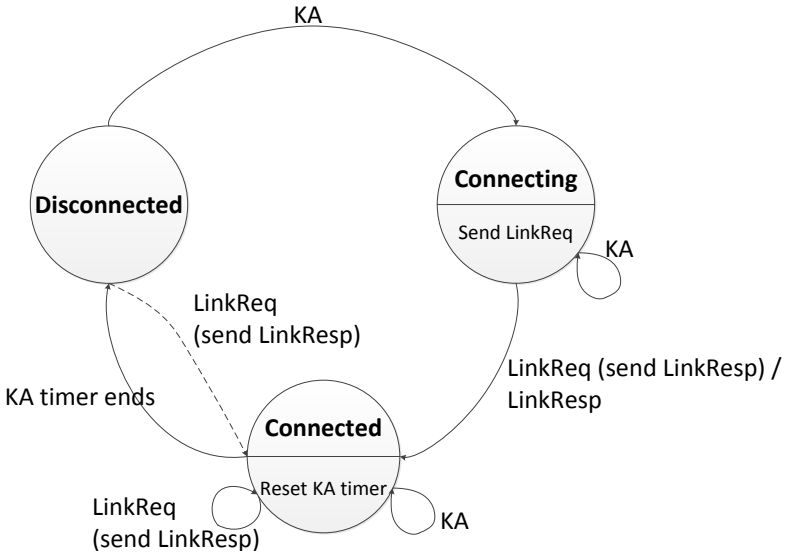

Figura 2.3: *FSM* do processo de descoberta.
[1]

O plano de dados, encarregue da distribuição de mensagens táticas entre o pelotão, é independente do plano de controlo, uma vez que ambos podem funcionar em simultâneo. Quando um nó quer enviar uma mensagem tática para todos os elementos da rede, este difunde-a apenas uma vez. Os nós vizinhos, por sua vez, ao receberem a mensagem, retransmitem-na apenas uma vez e assim por diante. De notar que, a retransmissão de uma mensagem tática por parte de um nó que só tem um vizinho, apenas irá utilizar recursos da rede para notificar o nó que lhe enviou a mensagem em primeiro lugar. Assim sendo, nós com apenas um vizinho não retransmitem as mensagens táticas que recebem. Na Figura 2.4 pode observar-se um fluxograma do mecanismo que assegura a retransmissão única das mensagens táticas, por parte de cada nó da rede.

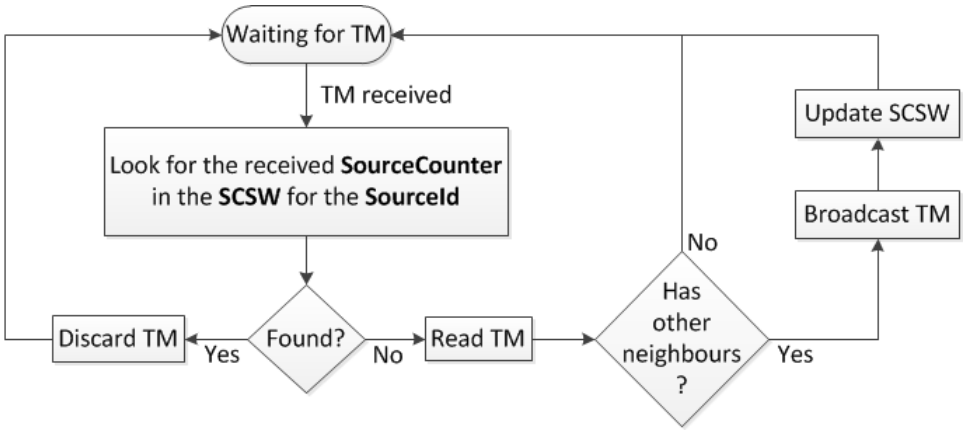

Figura 2.4: Mecanismo de retransmissão única.
[1]

Ao longo desta subsecção já foram abordados parcialmente os diferentes tipos de mensagens utilizados neste protocolo, sendo estes:

- *Keep Alive* - Utilizado para anunciar a presença do elemento emissor;

- *Link Request* e *Link Response* - Utilizados para sincronizar o processo de descoberta entre dois elementos;

- *Update* - Utilizado para partilhar o conhecimento da rede entre os diferentes elementos;

- *Tactical Messages* (TM) - Utilizado para partilha de dados entre todos os elementos da rede.

Na Figura 2.5, são apresentados os formatos dos tipos de mensagem referidos anteriormente.

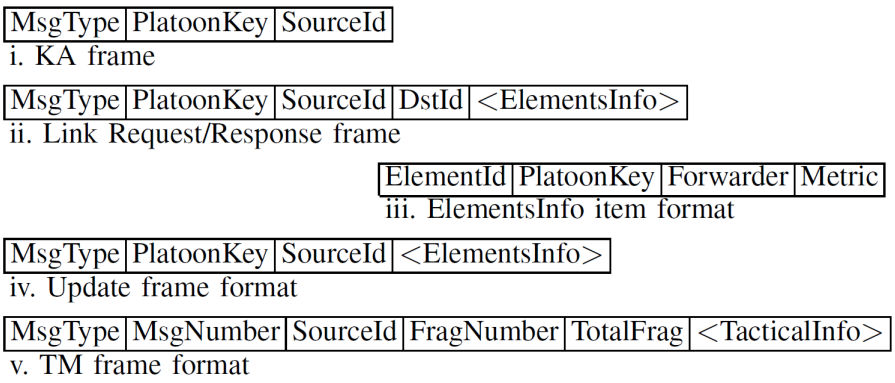

Figura 2.5: Formato das mensagens CRAN.
[1]

Os resultados apresentados por este protocolo, no que diz respeito à largura de banda necessária para manter o conhecimento da topologia da rede em cada nó de acordo com a realidade, demonstram que esta é reduzida e se encontra de acordo com a disponível em cenários militares. No que toca ao tempo entre a deteção de uma alteração na rede e a sua propagação por todos os nós da mesma, este é da ordem de grandeza das centenas de milissegundos, dependendo da topologia da rede, aumentando proporcionalmente de acordo com o número de nós. Contudo, este protocolo deixa a desejar no que toca à segurança, sendo possível que uma entidade maliciosa seja capaz de receber e enviar mensagens dentro da rede sem que esta seja detetada. Isto possibilita a fuga de informação e disseminação de contra-informação do/pelo pelotão, ambas tendo potencialmente consequências catastróficas, considerando uma operação a nível militar. Assim, uma resolução para este problema, que constitui o tema da presente dissertação, será abordada no capítulo 3.

Importa referir que, embora os resultados deste protocolo tenham sido obtidos a partir de testes realizados em cenários militares, este pode também ser utilizado em cenários civis.

2.2 Algoritmos de Cifra

Ao longo desta subsecção serão abordados dois tipos de cifra: cifra de chaves simétricas e cifra de chaves assimétricas. Por último, será feita uma comparação entre o desempenho dos diferentes algoritmos que implementam os tipos de cifra supracitados.

2.2.1 Cifras de chaves simétricas

A cifra de chaves simétricas baseia-se na utilização de duas chaves. Cada uma destas chaves é conhecida por um dos nós que pretendam comunicar, ou seja, um dos nós tem conhecimento de uma das chaves e o outro nó tem conhecimento da outra. Qualquer uma das chaves consegue decifrar uma mensagem que tenha sido cifrada pela outra. Note-se que, neste caso, a mesma chave não pode conseguir decifrar uma mensagem que tenha previamente cifrado. Importa ainda referir que uma chave é facilmente determinada a partir da outra, pelo que ambos os nós conseguem saber ambas as chaves. Porém, na maioria dos sistemas de criptografia ambas as chaves são iguais, permitindo que exista apenas uma chave que seja capaz de decifrar uma mensagem previamente cifrada pela própria, daí o termo de chave simétrica [2].

"A cifra é dita uma cifra de chaves simétricas se para cada par de chaves (e,d), é computacionalmente "fácil" determinar d sabendo só o valor de e e, de igual modo, determinar e de d." [2]

Quer sejam utilizadas duas chaves diferentes ou apenas uma partilhada pelos nós que pretendam comunicar, é importante que nenhum outro nó que não faça parte da comunicação em questão possa conhecê-las. Caso contrário, esta comunicação encontra-se comprometida. Daí existir dificuldade na distribuição das chaves entre os nós sem que estas tenham sido previamente combinadas, tendo de se recorrer a mecanismos de gestão de chaves [3].

As principais operações deste tipo de cifra são: ou-exclusivo, troca de colunas, troca de linhas, permutação, rotação e expansão. Estas operações caracterizam-se por terem um baixo custo computacional [4]. Os principais algoritmos que utilizam cifra de chaves simétricas são: *Data Encryption Standard* (DES) e *Advanced Encryption Standard* (AES) [2].

2.2.2 Cifras de chaves assimétricas

A cifra de chaves assimétricas baseia-se na utilização de duas chaves, chave pública (CP) e chave privada (CPriv). A chave pública é conhecida pelos nós da rede, enquanto que a chave privada é conhecida apenas pelo nó para o qual se pretende enviar a mensagem. Qualquer uma destas chaves consegue decifrar uma mensagem cifrada pela outra, contudo uma chave não consegue decifrar uma mensagem cifrada por si mesma. Assim, é possível que diferentes nós tenham conhecimento da chave pública, uma vez que apenas o nó que conhece a chave privada vai conseguir decifrar uma mensagem que lhe tenha sido enviada. Por essa razão, a chave privada não pode ser conhecida por mais nenhum nó, de modo a evitar que as mensagens sejam decifradas por outras entidades que não o nó de destino, comprometendo a confidencialidade e integridade da rede. É importante ainda que, a partir de uma das chaves, não seja possível obter a outra, pelas razões já referidas [4].

As principais operações deste tipo de cifra são o logaritmo discreto, curva elíptica e fatoração de inteiros. Estas operações acarretam um maior custo computacional do que as operações utilizadas nas chaves simétricas [4]. Os algoritmos de chave simétrica mais utilizados são: *Rivest-Shamir-Adleman* (RSA) e *Elliptic Curve Cryptography* (ECC).

Pelo uso deste tipo de cifra é possível uma segura distribuição de chaves entre nós e a assinatura das mensagens, algo que não é praticável em redes que utilizam chaves simétricas [4].

De acordo com [5], para o mesmo nível de segurança, o algoritmo ECC utiliza parâmetros mais pequenos em relação ao RSA. Estes parâmetros mais reduzidos traduzem-se em vantagens no que diz respeito à velocidade de processamento das cifras e no tamanho das chaves e certificados. Esta característica foi tida em conta e revelou-se importante na projeção do sistema de segurança proposto no capítulo 3.

2.2.3 Comparação do desempenho dos diferentes algoritmos de cifra

O ataque mais simples existente a um algoritmo de cifra é o ataque de força bruta, onde são percorridas exaustivamente todas as combinações possíveis de chave para o algoritmo em questão. Deste modo, a segurança de um algoritmo de cifra contra este tipo de ataques é tanto maior quanto maior forem as combinações de chaves existentes. Para diferentes algoritmos de chaves simétricas apresenta-se, na tabela 2.1, o respetivo número de chaves possíveis.

Tabela 2.1: Número de chaves possíveis para diferentes algoritmos de chave simétrica.

Algoritmo	DES	TDEA	AES (128 bits)	AES (192 bits)	AES (256bits)
Nº de chaves possíveis:	2^{56}	2^{90}	2^{128}	2^{192}	2^{256}

O tempo necessário para descobrir a chave de um destes algoritmos depende diretamente, não só do número de combinações de chaves possíveis, mas também do tempo necessário para testar cada chave.

Segundo [2], o sistema DES é suscetível a criptoanálise diferencial (CAD) e linear (CAL). Ataques deste tipo são mais eficientes, pois requerem um menor número de chaves a testar do que um ataque de força bruta, 2^{47} e 2^{43}, respetivamente. Em 1993, já era possível construir um sistema capaz de descobrir uma chave DES em cerca de 36 horas, com um custo de cem mil dólares. O sistema AES, devido às transformações que são efetuadas ao longo do algoritmo, não é suscetível a CAD nem CAL.

No que diz respeito às cifras de chave assimétrica, o algoritmo para quebrar chaves RSA mais utilizado e tido em conta como o mais eficiente é o *General Number Field Sieve* (GNFS). Este algoritmo baseia-se na fatoração de inteiros e, através deste, em 2003 foi possível descobrir uma chave RSA (576 bits). Prevê-se ainda que, no ano 2018, seja possível descobrir uma chave RSA (1024 bits). Em relação a chaves ECC, os algoritmos mais eficientes baseiam-se no cálculo de logaritmos discretos. Em 2002 conseguiu descobrir-se uma chave ECC (109 bits).

2.3 Assinatura Digital

A assinatura digital é um conceito criptográfico fundamental para garantir a autenticação, integridade e não-repúdio de uma mensagem. Este processo fornece um meio de associar determinada mensagem à entidade que a gerou [2].

A assinatura é criada através da transformação da mensagem que se pretende assinar e determinada informação secreta (por norma, chave privada), que apenas é do conhecimento da entidade que a gerou, impedindo assim que outras entidades possam falsificar essa assinatura. Por fim, o par assinatura e mensagem é enviado. Deste modo, qualquer entidade que receba este par consegue obter a mensagem com garantia de autenticidade, integridade e não-repúdio da mesma. Para o fazer, a entidade recetora da mensagem terá de executar um processo de verificação da assinatura, processo este que é público. Este processo está, por norma, associado à chave pública da entidade emissora [2].

A utilização desta primitiva será bastante importante na definição da segurança do protocolo CRAN (capítulo 3), pois é através dela que é possível garantir os conceitos de segurança referidos anteriormente (autenticação, integridade e não-repúdio).

2.4 Certificados

De acordo com [6],os certificados são documentos estruturados de forma predefinida e que contêm informações relativas a determinada entidade. Estes certificados são emitidos por uma Entidade Certificadora (*Certification Authority, CA*).

Os certificados têm validade limitada, sendo que essa validade pode ser controlada através dos prazos de validade indicados no próprio certificado ou através da emissão de certificados de revogação, emitidos geralmente pela *CA*. Estes certificados de revogação indicam expressamente que determinada chave pública pertencente a uma entidade deixa de ser válida a partir de uma data específica, por norma, antes do prazo de validade, contido no certificado, expirar.

A estrutura dos certificados encontra-se de acordo com a norma X.509, que define o seu formato base, ou seja, as suas informações e o formato como estas são organizadas. Dentro desta norma existe um conjunto de informações obrigatórias, sendo estas:

- Número da versão - Indica qual a versão do formato X.509, de acordo com a qual o certificado está organizado;
- Número de série - Número único para cada certificado emitido pela mesma *CA*. Por consequência, o par número de série e nome da *CA* é único;
- Algoritmo de assinatura - Algoritmo utilizado para gerar a assinatura do certificado. Regra geral, são indicados dois algoritmos, um de síntese (*Hash*) e outro de cifra com uma chave privada;
- *Issuer Name* - Identificação da *CA*;
- Validade do certificado - Dividido em 2 campos, um com a data e hora do início da validade e outro com o fim;
- *Subject Name* - Identificação da entidade a quem a chave pública presente no certificado pertence;

- Informação de chave pública - Dividido em 2 campos, a chave pública certificada e o respetivo algoritmo.

Tendo em consideração que o sistema projetado na presente dissertação deverá seguir um compromisso entre segurança e largura de banda necessária, os certificados de acordo com a norma X.509 têm, inevitavelmente, um tamanho elevado. Isto obriga a que as mensagens transmitidas na rede que contenham os certificados possam comprometer o funcionamento do sistema. Assim sendo, procurou-se a utilização de tipos de certificados com tamanho reduzido.

Em [7] são apresentados vários modelos de certificados de tamanho significativamente mais reduzido em comparação com a norma X.509. Embora nenhum desses modelos apresente a capacidade para introduzir informação opcional, é possível retirar a ideia base do seu funcionamento e aplicá-la ao sistema que se pretende projetar. Esta ideia assenta na existência de uma cadeia de certificados, no qual o certificado da AC representa o nível mais alto sendo, portanto, um certificado de maior tamanho do que os de níveis inferiores. Isto porque a informação relativa aos algoritmos utilizados, quer para cifrar ou assinar, se encontra nesse certificado. Posteriormente, os certificados de níveis inferiores não necessitam suportar essas informações desde que os algoritmos a utilizar sejam os mesmos, tornando-os mais pequenos. Importa referir que, para que esta ideia seja vantajosa em detrimento da norma X.509, o certificado da AC não pode ser enviado pela rede, pelo que cada nó da rede deverá conter este certificado *a priori*. Outro dos conceitos desta ideia baseia-se na utilização destes certificados apenas dentro do domínio da rede, o que proporciona uma maior redução do seu tamanho, pois deixa de existir a necessidade de uma identificação única universal. Apenas é necessário identificar de forma única, as entidades que fazem parte da rede. A implementação desta ideia de acordo com as características da rede é explicada mais detalhadamente na subsecção 3.2.3.

2.5 Autoridade Certificadora

Tipicamente, a gestão de chaves em sistemas com cifra de chaves assimétricas é feita com o auxílio de uma entidade confiável, denominada Autoridade Certificadora (AC), que utiliza um mecanismo de chave assimétrica, onde a sua chave pública é conhecida por todos os nós. Esta AC emite certificados, por meio da sua assinatura digital, que permitem comprovar a autenticidade de um nó (um certificado permite autenticar um nó). Um certificado associa o nó que se pretende autenticado à sua chave pública, permitindo aos outros nós que receberam esse certificado decifrar as mensagens enviadas pelo anterior e ter a certeza de que é realmente quem diz ser [8]. Este método garante, ainda, a integridade e não-repúdio, além de autenticação.

Esta gestão necessita que a AC se mantenha constantemente online, uma vez que apenas esta pode emitir os certificados. Caso contrário, os nós da rede não têm como receber os certificados e não conseguem estabelecer uma comunicação segura com outros nós. A AC torna-se, assim, um ponto crítico da rede, uma vez que, caso seja comprometida e seja descoberta a sua chave privada, facilmente uma entidade maliciosa consegue emitir certificados falsos para personificar um outro nó ou revogar certificados anteriormente atribuídos pela AC [9].

Uma abordagem que permite o aumento da disponibilidade consiste na existência de réplicas da AC. No entanto, existe uma maior vulnerabilidade do sistema, pois basta uma das réplicas ser comprometida para que seja possível acontecerem os problemas que foram expostos no parágrafo anterior [9].

Para resolver as vulnerabilidades apresentadas anteriormente, foi proposto um mecanismo onde não existam ACs, que consiste em que os próprios nós emitam certificados uns aos outros[10].

2.6 Gestão de Chaves

Na presente subsecção, irão ser abordados conceitos pertinentes à gestão de chaves em redes fixas e Ad-hoc móveis. Conceitos estes que se revelaram importantes na aquisição de conhecimentos acerca desta área, mesmo que na sua grande maioria não tenham sido utilizados na elaboração do sistema de segurança. Isto deve-se à especificidade do sistema que se quer implementar, uma vez que se pretende que opere principalmente em ambiente militar.

2.6.1 Gestão de Chaves em Redes Fixas

Para garantir a segurança das redes, utilizam-se algoritmos criptográficos. Estes algoritmos, partem do princípio que a distribuição de chaves pela rede é feita de forma adequada, de modo a evitar falhas de segurança na mesma. Esta distribuição de chaves pela rede tem o nome de gestão de chaves. Ao longo desta subsecção, irão ser abordadas soluções que possibilitam a autenticação, integridade, confidencialidade, controlo de acesso e não-repúdio.

De seguida, serão apresentados dois sistemas de geração de chaves simétricas, sendo que o primeiro gera uma chave apenas entre duas entidades e o segundo gera-a entre um grupo de entidades.

Diffie-Hellman

Diffie-Hellman é um sistema de distribuição de chaves simétricas proposto por [11], necessitando que apenas uma chave seja trocada e permite que essa troca possa ser feita através de um canal público onde uma terceira entidade possa estar à escuta, sem que consiga descobrir a chave partilhada.

Inicialmente é acordado, entre os 2 utilizadores, i e j, que pretendem comunicar e através do canal público, um valor para α e outro para q, sendo que q terá de ser um número primo e α um número que, quando elevado a qualquer outro número inteiro entre 1 e q-1 (inclusive), obtenha sempre resultados diferentes. Os valores de α e q podem ser acordados previamente. No entanto, partindo do princípio que ambos os utilizadores nunca tiveram contacto um com o outro, estes valores podem ser partilhados através de um canal público, sem que o conhecimento destes por parte de terceiros comprometa o sistema.

De seguida, cada utilizador, i e j, gera um número aleatório X_i e X_j, respetivamente. Estes números terão de ser entre 1 e q-1 (inclusive) e não são partilhados no canal, sendo mantidos secretos. Com esses números, X_i e X_j, cada utilizador irá fazer os respetivos cálculos definidos pelas expressões

(2.1) e (2.2). Esses resultados, são colocados num ficheiro público, apenas de leitura, junto com o seu nome e endereço (um ficheiro para cada resultado).

$$Y_i = \alpha^{X_i} \pmod{q} \tag{2.1}$$

$$Y_j = \alpha^{X_j} \pmod{q} \tag{2.2}$$

Deste modo, quando os 2 utilizadores pretenderem comunicar em privado, geram uma chave calculada com base na expressão (2.3).

$$K_{ij} = \alpha^{X_i X_j} \pmod{q} \tag{2.3}$$

Para o fazer, não é necessário que os utilizadores saibam os números secretos um do outro (X_i e X_j). Uma vez que é possível obter Y_i e Y_j a partir do ficheiro público, é possível calcular a expressão (2.3) com base na expressão (2.4).

$$\begin{aligned} K_{ij} &= Y_j^{X_i} \ mod \ q \\ &= (\alpha^{X_j})^{X_i} \ mod \ q \\ &= \alpha^{X_j X_i} \ mod \ q \\ &= \alpha^{X_i X_j} \ mod \ q \end{aligned} \tag{2.4}$$

Uma terceira entidade que queira obter a chave de comunicação entre i e j, terá de calcular (2.5), uma vez que não tem acesso a X_i nem X_j.

$$K_{ij} = Y_i^{(log_\alpha Y_j)} \pmod{q} \tag{2.5}$$

No entanto, é computacionalmente impossível fazê-lo, ao contrário de 2.3 que é computacionalmente fácil de calcular, como é explicado no parágrafo seguinte.

O cálculo da expressão (2.3) necessita no máximo de $2 * b$ multiplicações, enquanto que o cálculo da expressão (2.5) necessita de $2^{b/2}$ operações. Para determinar o valor de b, este deverá ser o menor inteiro que consiga garantir a condição $2^b > q$, ou seja:

$$\begin{cases} min(b) \\ 2^b > q \end{cases} \tag{2.6}$$

Assim sendo, o esforço computacional exercido para calcular a expressão (2.5) cresce de modo exponencial relativamente ao cálculo da expressão (2.3). Por exemplo, para calcular 2.3 para b=128, serão necessárias no máximo 256 multiplicações, já para o cálculo de 2.5 serão necessárias 2^{64} operações. De notar que a segurança deste sistema depende diretamente da dificuldade computacional em calcular logaritmos de modulo q.

Na tabela 2.2 encontram-se expostas as vantagens e desvantagens deste algoritmo.

Tabela 2.2: Diffie-Hellman: vantagens e desvantagens.

Vantagens	Desvantagens
Apenas é necessário que uma chave seja trocada.	Caso logaritmos de módulo se tornem computacionalmente fáceis de calcular, o sistema pode ser comprometido.
O esforço para que uma terceira entidade descubra a chave cresce exponencialmente, relativamente ao esforço necessário para utilizadores legítimos.	Neste processo não está prevista a autenticação entre os elementos que contribuem para gerar a chave.
A existência de *perfect forward secrecy* (*PFS*), uma vez que, a cada comunicação com diferentes elementos, é gerada uma nova chave de sessão.	

Diffie-Helman: distribuição de chaves em grupo

De seguida, serão apresentados sistemas de distribuição de chaves em grupo, propostos por [12], com base no sistema Diffie-Hellman.

A primeira proposta de [12] consiste em 2 fases: ascendente e descendente. A fase ascendente tem como objetivo adquirir as contribuições de todos os elementos do grupo, através do calculo de um expoente e envio deste e de outros valores intermédios para o membro seguinte. Para o fazer, o membro M_i recebe um conjunto de valores intermédios de M_{i-1}, calcula $\alpha^{N_1 \dots N_i}$ elevando $\alpha^{N1 \dots Ni-1}$ ao expoente N_i e junta-o aos valores intermédios recebidos, enviando-os para o membro seguinte M_{i+1}. Sendo que N_i é um número gerado aleatoriamente pelo membro M_i. Quando o último membro M_n, recebe os valores intermédios, este calcula do mesmo modo que os restantes membros $(\alpha^{N_1 \dots N_{n-1}})^N_n$, este valor calculado será a chave de grupo, K_n. A fase descendente é iniciada em M_n e tem como objetivo distribuir K_n pelos restantes membros do grupo. Assim, M_i retira o último dos valores intermédios, eleva-o a N_i, obtendo deste modo K_n. Após isto, todos os outros valores intermédios são igualmente elevados a N_i e, seguidamente, enviados para M_{i-1}. Este processo é executado até que seja atingido M_1, garantindo assim que todos os membros do grupo obtenham K_n. Deste modo, é também evitado, pelos princípios de Diffie-Hellman, que terceiros, que se encontrem à escuta na rede, obtenham K_n.

Para um melhor esclarecimento desta proposta, é possível observar em 2.7 e 2.8 um exemplo da fase ascendente e descendente deste sistema, respetivamente. Supondo que existem 5 membros pertencentes ao grupo, as mensagens enviadas de cada um dos membros, M_i, para os seguintes, M_{i+1}, no sentido ascendente são:

$$M_1 = \{\alpha^{N_1}\}$$

$$M_2 = \{\alpha^{N_1}, \alpha^{N_1 N_2}\}$$

$$M_3 = \{\alpha^{N_1}, \alpha^{N_1 N_2}, \alpha^{N_1 N_2 N_3}\} \tag{2.7}$$

$$M_4 = \{\alpha^{N_1}, \alpha^{N_1 N_2}, \alpha^{N_1 N_2 N_3}, \alpha^{N_1 N_2 N_3 N_4}\}$$

As mensagens enviadas de cada um dos membros, M_i, para os anteriores, M_{i-1}, no sentido descendente são:

$$M_5 = \{\alpha^{N_5}, \alpha^{N_1 N_5}, \alpha^{N_1 N_2 N_5}, \alpha^{N_1 N_2 N_3 N_5}\}$$

$$M_4 = \{\alpha^{N_4 N_5}, \alpha^{N_1 N_4 N_5}, \alpha^{N_1 N_2 N_4 N_5}\}$$

$$M_3 = \{\alpha^{N_3 N_4 N_5}, \alpha^{N_1 N_3 N_4 N_5}\} \tag{2.8}$$

$$M_2 = \{\alpha^{N_2 N_3 N_4 N_5}\}$$

Com o objetivo de reduzir o número de rondas e mensagens necessárias para gerar e distribuir uma chave por todo o grupo, foi proposto um segundo sistema, que consiste também numa fase ascendente e numa fase descendente, como na primeira proposta de [12]. A fase ascendente, tal como na primeira proposta, tem como objetivo adquirir as contribuições de todos os elementos do grupo, no entanto a forma de o fazer é diferente. Cada M_i tem de gerar valores intermédios, que são obtidos realizando $i-1$ potenciações, tendo como base os valores intermédios recebidos de M_{i-1} e como expoente N_i. Entre estes valores intermédios gerados por M_i, encontra-se $\alpha^{N1...Ni}$, o valor cardinal. A este conjunto de valores intermédios a serem enviados para M_{i+1} é ainda adicionado o valor cardinal de M_{i-1}, $\alpha^{N_1...N_{i-1}}$. Por fim, quando M_n gerar os seus valores intermédios, o seu valor cardinal será a chave de grupo K_n. Na fase descendente, M_n envia os seus valores intermédios para todos os outros membros do grupo, permitindo que estes ao exponenciar o valor intermédio correspondente a cada um, ao expoente N_i, conseguem obter K_n. Mais uma vez, os princípios de Diffie-Hellman foram cumpridos, uma vez que os valores de $N_1...N_n$ e K_n nunca foram enviados pela rede de forma a que terceiros os consigam obter. Com esta proposta, consegue reduzir-se o número de rondas e mensagens necessárias para cerca de metade, em relação à primeira proposta.

Para um melhor esclarecimento desta proposta, é possível observar em 2.9 e 2.10 um exemplo da fase ascendente e descendente deste sistema, respetivamente. Supondo, mais uma vez, que existem 5 membros pertencentes ao grupo, as mensagens enviadas de cada um dos membros, M_i, para os seguintes, M_{i+1}, no sentido ascendente são:

$$M_1 = \{\alpha^{N_1}\}$$

$$M_2 = \{\alpha^{N_1 N_2}, \alpha^{N_1}, \alpha^{N_2}\}$$

$$M_3 = \{\alpha^{N_1 N_2 N_3}, \alpha^{N_1 N_2}, \alpha^{N_1 N_3}, \alpha^{N_2 N_3}\} \tag{2.9}$$

$$M_4 = \{\alpha^{N_1 N_2 N_3 N_4}, \alpha^{N_1 N_2 N_3}, \alpha^{N_1 N_2 N_4}, \alpha^{N_1 N_3 N_4}, \alpha^{N_2 N_3 N_4}\}$$

Note-se que os primeiros valores de cada mensagem são os valores cardinais.

A única mensagem do sentido descendente é enviada por M_5 para todos os outros membros, M_i $i\epsilon[1, n-1]$:

$$M_5 = \{\alpha^{N_1N_2N_3N_5}, \alpha^{N_1N_2N_4N_5}, \alpha^{N_1N_3N_4N_5}, \alpha^{N_2N_3N_4N_5}\} \tag{2.10}$$

Um terceiro sistema foi proposto com o objetivo de reduzir o número de exponenciações realizadas por membro do grupo, face às duas propostas anteriores. Esta terceira proposta difere bastante das anteriores, consistindo em 4 fases. A primeira fase é idêntica à da primeira proposta, adquirindo contribuições de todos os membros do grupo. No entanto, esta fase termina assim que M_{n-1} receba $\alpha^{N_1....N_{n-2}}$ e calcule $\alpha^{N1....Nn-1}$. Na segunda fase, M_{n-1} encarrega-se de enviar $\alpha^{N_1...N_{n-1}}$ para todos os membros do grupo. Na terceira fase, cada membro M_i calcula N_i^{-1}, o inverso do seu próprio expoente, N_i, de modo a poder retirá-lo de $\alpha^{N_1....N_{n-1}}$ através de potenciação, calculando assim $\alpha^{N_1...N_{i-1}N_{i+1}...N_{n-1}}$, e envia esse resultado para M_n. Na quarta e última fase, M_n recebe os valores enviados por todos os outros membros do grupo, eleva todos esses valores a N_n e envia-os de volta para todos os membros. Deste modo, cada membro M_i irá receber de M_n $\alpha^{N_1...N_{i-1}N_{i+1}...N_n}$, podendo assim obter K_n, calculando $\alpha^{N_1...N_n} = (\alpha^{N_1...N_i-1N_{i+1}...N_n})^{N_i}$. Com esta proposta, consegue reduzir-se o número de exponenciações por membro do grupo para 4, exceto para o M_{n-1} e M_n, que são necessárias 2 e n exponenciações, respetivamente. Isto fará com que seja exigido menos poder computacional por parte de cada nó, à exceção de M_n. Outra das vantagens desta proposta é o tamanho constante das mensagens trocadas pelo grupo, ainda que a quantidade das mesmas duplique face à segunda proposta.

Para o mesmo número de membros pertencentes ao grupo, n=5, é possível observar em 2.11, 2.12, 2.13 e 2.14 exemplos das 4 fases deste sistema, respetivamente. Na primeira fase, as mensagens enviadas de cada um dos membros, M_i, para os seguintes, M_{i+1}, são:

$$\begin{aligned} M_1 &= \{\alpha^{N_1}\} \\ M_2 &= \{\alpha^{N_1N_2}\} \\ M_3 &= \{\alpha^{N_1N_2N_3}\} \end{aligned} \tag{2.11}$$

Note-se, neste exemplo, que assim que M_4 recebe a mensagem vinda do membro anterior, já não são enviadas mais mensagens nesta primeira fase. Na segunda fase, apenas é enviada uma mensagem por parte de M_4 para todos os outros membros M_i $i \in [1, n-2]$:

$$M_4 = \{\alpha^{N_1N_2N_3N_4}\} \tag{2.12}$$

Após os cálculos inerentes à terceira fase terem sido efetuados pelos membros M_i $i \in [1, n-2]$, todos estes, incluindo M_4, enviam as seguintes mensagens para M_5:

$$M_1 = \{\alpha^{N_2N_3N_4}\}$$

$$M_2 = \{\alpha^{N_1 N_3 N_4}\}$$

$$M_3 = \{\alpha^{N_1 N_2 N_4}\} \tag{2.13}$$

$$M_4 = \{\alpha^{N_1 N_2 N_3}\}$$

Na última fase, M_5 envia apenas uma mensagem para todos os outros membros do grupo, permitindo que estes possam, por fim, calcular K_n:

$$M_5 = \{\alpha^{N_2 N_3 N_4 N_5}, \alpha^{N_1 N_3 N_4 N_5}, \alpha^{N_1 N_2 N_4 N_5}, \alpha^{N_1 N_2 N_3 N_5}\} \tag{2.14}$$

Uma vez que na subsecção 2.6.2 é apresentado um sistema que melhor se enquadra com as necessidades onde a rede irá operar, os algoritmos abordados nesta subsecção não foram utilizados na proposta de segurança para o protocolo CRAN (capítulo 3). A carência de autenticação nestes algoritmos é outro dos fatores que leva à não utilização dos mesmos.

2.6.2 Gestão de Chaves em Redes Moveis Ad-Hoc

Em *Mobile Ad-hoc Networks* (MANETs) é importante ter em conta que um nó funciona como terminal e como *router* intermédio para reencaminhar o tráfego para outros nós. Assim sendo, é necessário considerar a existência de nós maliciosos que se façam passar por nós intermédios, cujo objetivo seja intercetar as comunicações realizadas, sendo pertinente evitar que tal aconteça. Para além das características que a gestão de chaves em grupo deve garantir à rede (autenticação, integridade, confidencialidade, controlo de acesso e não-repúdio), em MANETs é ainda necessário ter em consideração outros fatores, como a mobilidade dos nós, a largura de banda disponível e o consumo de bateria dos nós.

Para fazer face aos fatores referidos, foi proposto por [13] um sistema de distribuição de chaves que tem como ideia base a existência de um líder de grupo, que é o responsável pela gestão de chaves dentro do grupo. Isto pode ser interessante na resolução do problema proposto nesta dissertação, uma vez que o nó do CmdtPel (líder de grupo) poderá ficar a cargo das responsabilidades referidas. Assim sendo, cada pelotão poderá representar um grupo que tem como líder o seu Comandante. Além disso, este sistema não necessita da existência de uma AC, o que é outro fator importante na resolução do problema.

Neste sistema são definidos 4 tipos de chaves:

- Chave de grupo (chave simétrica) – Utilizada por todos os membros do grupo para cifrar e decifrar as mensagens partilhadas pelo grupo;
- Chave partilhada (chave simétrica) – Partilhada entre o líder do grupo e um membro desse mesmo grupo, no momento em que o segundo se junta ao grupo;
- Chave de líder – Partilhada por todos os líderes de grupo na rede, gerada por um sistema de gestão de chaves em grupo baseado em Diffie-Hellman, idêntico aos sistemas apresentados na subsecção 2.6.1.

- Chave pública/privada – Diferentes pares de chaves assimétricas (pública e privada), associados a cada nó. Utilizadas essencialmente para distribuição de novas chaves de grupo, após adição ou remoção de nós da rede.

Cada líder gera a chave de grupo para o seu respetivo grupo. Chave esta, que é atualizada cada vez que é adicionado ou removido um membro ao grupo, de modo a assegurar que o nó que tenha sido adicionado não tenha acesso às mensagens trocadas anteriormente pelo grupo, e que o nó que tenha sido removido não tenha acesso às futuras mensagens trocadas pelo grupo.

Em relação aos sistemas apresentados em 2.6.1, este sistema traz ainda a vantagem de que a chave de grupo não é recalculada em todos os nós. Em vez disso, é atualizada pelo líder e enviada para os outros nós do grupo, não exigindo esforço adicional por parte de todos os nós do grupo. Esta é outra característica importante para a resolução do problema proposto, uma vez que vai diminuir o consumo de energia dos nós pertencentes ao grupo (exceto do nó líder).

O sistema proposto na presente dissertação (capítulo 3) integra, em grande parte, as ideias apresentadas nas próximas duas subsecções (2.6.2 e 2.6.2). Isto deve-se ao facto da existência de um líder, indo de encontro às necessidades que o sistema apresenta, uma vez que o cenário onde irá operar é maioritariamente de natureza militar, onde a existência de um líder é crucial.

Adição de membros

Para evitar ataques do tipo *Man-in-the-middle*, sempre que um novo nó faz um pedido de adição ao grupo, é realizada uma autenticação mútua entre o líder do grupo e o nó que se pretende juntar ao grupo. Esta autenticação poderá ser feita através de um desafio lançado tanto ao líder do grupo, como ao nó que se pretende juntar ao grupo, e terá de ser cumprido por ambos. Infelizmente, o artigo [13] não é claro no esclarecimento de como este desafio é realizado.

Uma vez garantida a autenticação de ambos os nós, o que se pretende juntar à rede envia ao líder o seu certificado (emitido por uma entidade confiável, não presente na rede). O líder, por sua vez, verifica o certificado enviado pelo nó que se pretende juntar e daí obtém a sua chave pública. Seguidamente, o líder envia para o novo nó uma mensagem cifrada pela chave pública deste, contendo: a identificação (ID) atribuída ao novo nó dentro da rede e a chave partilhada entre ambos. Depois, atualiza a lista de membros do grupo e gera uma nova chave de grupo, enviando estes dados para o novo nó, cifrados pela chave partilhada. Finalizando este processo, o líder envia para todos os nós pertencentes ao grupo uma mensagem cifrada pela antiga chave de grupo, contendo a nova chave de grupo, bem como a lista de membros do grupo atualizada.

Na Figura 2.6 encontra-se um esquema do funcionamento do processo explicado acima. Como é possível observar, a existência de um líder que controla o acesso dos nós ao grupo integra-se na ideia que se pretende para o sistema desenvolvido no capítulo 3.

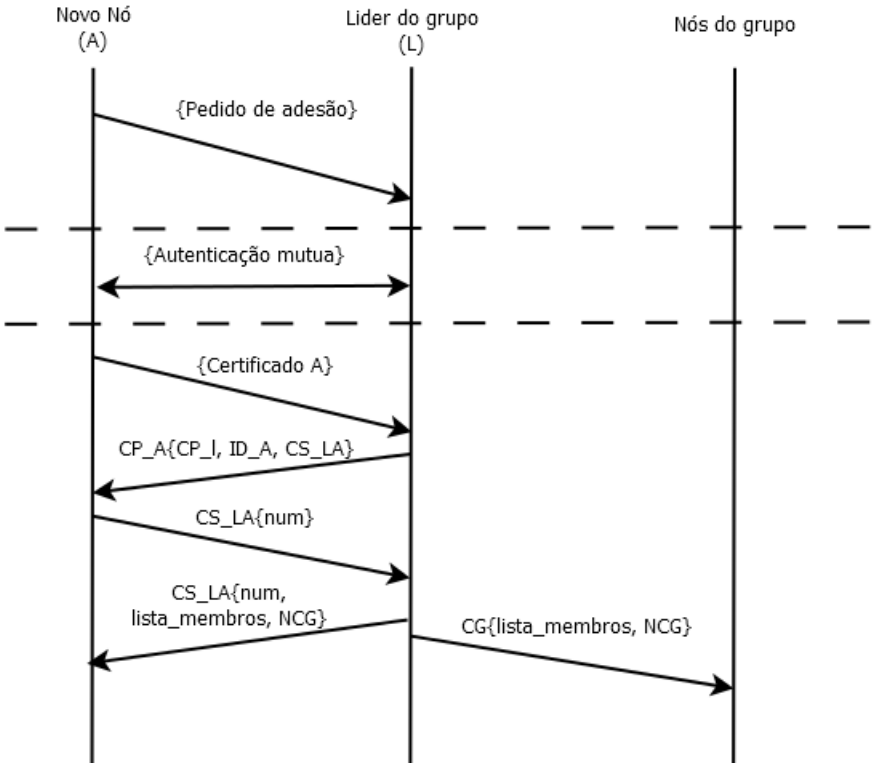

Figura 2.6: Adição de novo nó.

Remoção de membros

Para remover um membro do grupo, o líder gera uma nova chave de grupo e uma lista de membros atualizada (sem o nó que se pretende remover), enviando ambas numa mensagem para cada um dos nós do grupo (excluindo o nó removido), cifrada pela chave pública do respetivo nó recetor.

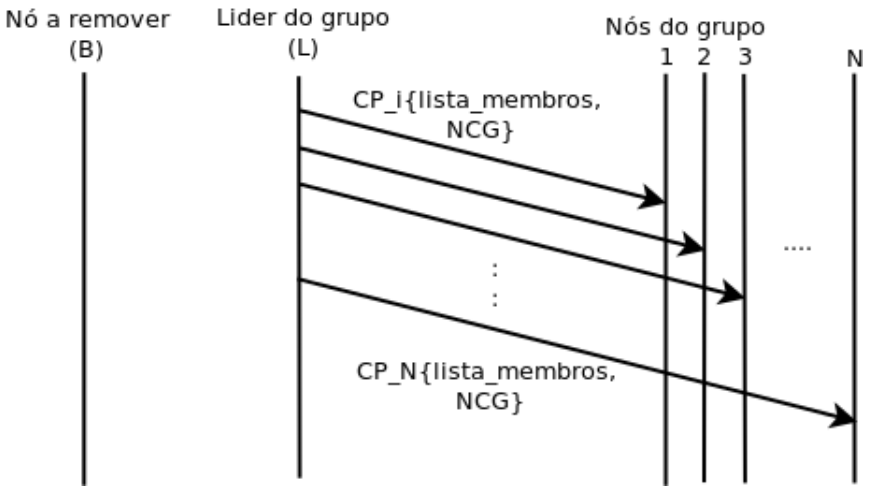

Figura 2.7: Remoção de nó da rede.

Um nó poderá ser removido do grupo caso deixe de comunicar a sua presença no grupo ao líder ou notifique previamente a sua saída do grupo. Existe ainda outro motivo para a remoção de um nó do grupo: quando este não reencaminhe as mensagens ou aja de modo idêntico a um nó malicioso, levando à sua remoção do grupo e posterior aviso aos líderes de outros grupos, por parte do líder do

grupo de onde foi removido. Na Figura 2.7 encontra-se um esquema do funcionamento do processo explicado acima.

Capítulo 3

Proposta de segurança para o CRAN

O sistema proposto foi projetado tendo em consideração as características de uma rede móvel *ad-hoc* com largura de banda reduzida. Este sistema tem como objetivo assegurar a confidencialidade, integridade, autenticação, proteção contra replicação e não-repúdio das mensagens utilizadas pelo protocolo CRAN, bem como a autenticação dos nós pertencentes à rede na qual este se insere. A ideia fundamental é, portanto, utilizar o mínimo de largura de banda possível sem comprometer a segurança. Para tal, foram criados 5 tipos de mensagem (*Join Request*, *Join Reply*, *Leave Request*, *Leave Reply* e *Key Refresh*, cuja explicação se encontra nas subsecções 3.4.1, 3.4.2, 3.5.1, 3.5.2 e 3.6, respetivamente) e ainda acrescentados campos a um dos tipos de mensagens já existentes no protocolo CRAN (KA, explicado na subsecção 3.3).

Foi, ainda, tido em conta o tipo de chaves necessárias para garantir o funcionamento do sistema, sendo estas descritas em 3.2.7. Outros itens, como por exemplo identificadores, patentes e certificados, serão apresentados ao longo deste capítulo, bem como o seu contributo para o funcionamento do sistema em segurança.

O modo como foi projetado este sistema teve em consideração o funcionamento hierárquico das Forças Armadas (FA), sendo, por isso, restringido à existência de um líder de grupo. Contudo, foi delineado um processo para a eleição deste líder que se baseia numa hierarquia previamente designada, assemelhando-se assim ao processo de eleição de líder nas FA, em que por norma o comandante é o elemento de mais alta patente. Este processo tem a capacidade para eleger um novo líder ao longo da operação, caso por algum motivo, o líder anterior se encontre fora do grupo ou seja neutralizado. Deste modo, existe sempre uma entidade responsável pelo grupo, não comprometendo o funcionamento da rede. Ainda neste contexto, a quebra de ligação entre um determinado subgrupo de elementos poderá ocorrer, seja este resultado de um afastamento excessivo entre os nós ou da presença de um obstáculo que impeça a comunicação. Esta separação em 2 ou mais subgrupos leva a que o processo de eleição de líder referido anteriormente entre em ação para os subgrupos nos quais o líder inicial não se encontra presente.

O nó que for o atual líder de grupo durante a utilização da rede irá desempenhar um papel fulcral no funcionamento deste sistema, uma vez que é este que tem a capacidade para aceitar pedidos de

adesão de outros nós ao grupo, ou de removê-los do mesmo. É ainda o responsável pela atualização da chave de grupo cada vez que o número de elementos no mesmo se altere. Alguns dos detalhes de funcionamento deste sistema têm características idênticas ao sistema abordado em 2.6.2.

Este sistema oferece, através dos seus tipos de mensagens, uma camada de segurança ao protocolo CRAN. Consequentemente, para envio e receção das mensagens previstas no CRAN, entre elementos pertencentes a determinado grupo, basta cifrá-las com a chave de grupo. Assim, toda a troca de mensagens de acordo com o protocolo CRAN pode ser realizada sob a segurança de uma cifra, mantendo a confidencialidade das mensagens.

3.1 Modelo do Atacante

Para uma análise das características de segurança do sistema proposto, revela-se necessária a definição de um modelo da entidade atacante, que visa expor as capacidades da mesma no que diz respeito a ataques contra a rede. No modelo de proposto por [14], existem dois tipos de participantes numa rede: o participante legítimo e o adversário ou atacante. O participante legítimo, cumpre sempre o funcionamento do protocolo sem alterações e pode participar em várias instâncias deste com outros participantes. O atacante, por sua vez, irá agir de modo arbitrário na tentativa de atingir o seu objetivo, não sendo obrigado a seguir o funcionamento do protocolo.

Apoiando neste modelo e em [15], devem assumir-se as seguintes características em relação ao atacante:

- O atacante consegue obter todas as mensagens que são trocadas na rede;
- O atacante consegue iniciar uma troca de mensagens entre outros nós;
- O atacante pode ser o recetor de qualquer emissor;
- O atacante não tem limitações de energia;
- O atacante tem conhecimento do protocolo utilizado na rede;
- Tem conhecimento das movimentações dos nós na rede e dos respetivos vizinhos.

Em relação às limitações apresentadas pelo atacante, este não tem capacidade para realizar as seguintes tarefas:

- Cifrar e decifrar uma mensagem sem conhecimento da chave necessária para o fazer;
- Determinar a chave privada de um nó, a partir da sua chave pública;
- Manipular diretamente os nós. Apenas consegue manipular as comunicações.

Segundo o modelo de Dolev-Yao, deve considerar-se que qualquer mensagem enviada pela rede foi enviada pelo atacante. Assim sendo, qualquer mensagem poderá ter sido manipulada e é necessário um sistema de segurança que permita aos nós legítimos da rede, descartar estas mensagens enviadas pelo atacante.

Tendo em consideração que a rede em questão é uma rede móvel Ad-hoc, facilmente o atacante consegue escutar, manipular, injetar e duplicar uma mensagem, uma vez que este é considerado como presente em toda a rede. O objetivo é, então, impedir que o atacante comprometa a rede, negando-lhe acesso às mensagens trocadas entre os nós da rede e evitando que este envie mensagens para a rede.

Posto isto, há que considerar o atacante em duas vertentes distintas: uma em que este é um nó externo à rede e outra em que o atacante é um nó pertencente à rede que poderá ter sido comprometido. No primeiro caso, embora o atacante possa ter conhecimento do modo como funciona o sistema, este não tem acesso às chaves utilizadas na rede. No segundo caso, o atacante poderá ter acesso às chaves conhecidas pelo/s nó/s comprometido/s. Considerando as duas vertentes do atacante, este poderá então adquirir conhecimento das chaves públicas, das chaves privadas dos nós comprometidos e da identificação dos nós participantes na rede. Este conhecimento, associado ao conjunto de mensagens trocadas na rede, irá influenciar o modo como o adversário ataca a rede.

É necessário então fazer uma avaliação destas duas situações, aquando da projeção do sistema de segurança, de modo a evitar que o atacante consiga comprometer a segurança da rede, quer na escuta de informação, quer na injeção de informação prejudicial.

É importante, ainda, considerar o modelo de ameaça STRIDE (*Spoofing*, *Tampering*, *Repudiation*, *Information disclosure*, *Denial of service*, *Elevation of privilege*) apresentado por [16], no qual são tidos em conta os diferentes tipos de ameaça que podem ser feitos a um sistema, permitindo uma abordagem correta aos mesmos, por forma a identificar e evitar o sucesso de ataques ao sistema que se pretende projetar. As ameaças definidas pela sigla STRIDE são neutralizadas garantindo autenticação, integridade, não-repúdio, confidencialidade, disponibilidade e autorização, respetivamente. Estas propriedades terão de ser implementadas no sistema, de modo a garantir que este esteja protegido contra as ameaças apresentadas pelo STRIDE. De acordo com [17], o modo como o modelo STRIDE deve ser aplicado é identificando como cada uma destas ameaças afeta as diferentes partes do sistema. Isto é, para cada um dos componentes do sistema, deve ser analisada a sua suscetibilidade às ameaças STRIDE que podem ocorrer, mitigando-as.

3.2 Definições e Conceitos Básicos

Nesta subsecção são abordados algumas definições e conceitos básicos que serão importantes ter em mente ao longo de todo o capítulo, para uma melhor perceção do funcionamento do sistema e do modo como foram solucionados os desafios do mesmo.

3.2.1 Identificador

Cada nó presente neste sistema tem um identificador (ID), que é atribuído pela autoridade certificadora no momento de emissão do certificado. Este ID é único para cada nó. Como no protocolo CRAN já estava previsto a utilização de um ID para cada nó, decidiu-se utilizar este para as funcionalidades acrescentadas. Deste modo, evita-se a utilização de dois IDs por cada nó (um ID previsto inicialmente

no CRAN e outro para a segurança).

3.2.2 Patente

À semelhança do identificador, cada nó tem uma patente. A patente é utilizada para estabelecer uma hierarquia entre os nós e, como tal, deve ser também única para cada elemento. Isto é importante para que seja possível eleger um líder de grupo dinamicamente ao longo da operação da rede, prevenindo a falha da rede num cenário em que o líder seja desabilitado ou numa partição do pelotão. Assemelha-se, portanto, ao cenário presente nas FA em que, por norma, o líder é eleito de acordo com a sua patente, daí o nome atribuído.

3.2.3 Certificado

De acordo com a ideia apresentada na subsecção 2.4, o sistema de certificados irá consistir numa cadeia de certificados de dois níveis, na qual o certificado da AC representa o nível mais alto. Os certificados de nível inferior, doravante denominados certificados subalternos, serão os certificados pertencentes a cada um dos nós que participem no sistema.

Durante o funcionamento deste sistema, será necessário emitir com frequência os certificados subalternos em conjunto com alguns tipos de mensagens. Assim sendo o ideal será reduzir ao máximo o tamanho dos mesmos. Para tal, e seguindo a ideia explanada em 2.4, o certificado da AC deverá conter toda a informação de acordo com o formato X.509. Posteriormente, os certificados subalternos poderão conter menos informação, uma vez que a informação relativa a vários campos já se encontra presente no certificado da AC, a saber:

- Algoritmo de assinatura - Este será o algoritmo de assinatura utilizado por toda a rede;
- Algoritmo de cifra assimétrica - Mais uma vez, este algoritmo será utilizado por toda a rede;
- Validade do certificado - Data e hora do início e fim da validade para todos os certificados (da AC ou subalternos).

Posto isto, os certificados subalternos irão conter apenas os seguintes campos:

- ID do nó - Em contraste com o *Serial Number* e com o *Subject Name*, uma vez que estes certificados só existem dentro da rede, apenas existe a necessidade de identificação única dentro da rede;
- Chave pública - Dividido em 2 campos, chave pública para cifra e outra para validar a assinatura, ambas certificadas;

O campo *Issuer Name* torna-se obsoleto, uma vez que este tipo de certificados são utilizados apenas dentro do domínio desta rede e existe apenas uma AC. Do mesmo modo, o campo, número da versão, é desnecessário pois apenas foi projetada uma versão para este tipo de certificados.

Outra das razões para a utilização de certificados subalternos é a necessidade de adicionar informação relativa aos nós, pertinente ao funcionamento da rede. Assim sendo, os certificados subalternos deverão conter ainda a informação relativa à patente do nó respetivo.

Importa referir que o certificado da AC deverá fazer parte integrante de cada nó *a priori*, e não poderá ser emitido pela rede, caso contrário, toda a premissa da existência de certificados subalternos é inválida.

3.2.4 Autoridade certificadora

Na projeção deste sistema decidiu-se que a AC deveria encontrar-se *offline*, isto porque a sua presença na rede torná-la-ia um alvo remunerador para o inimigo. Importa relembrar que um dos objetivos é fazer com que a rede tenha capacidade de se adaptar à possível destruição de elementos. Assim sendo, ao ter a AC *online*, estar-se-ia a falhar para com essa premissa, visto que a perda deste elemento centralizador comprometeria o funcionamento do sistema. Bastaria que o inimigo conseguisse neutralizá-la ou apoderar-se dela para comprometer o funcionamento ou a segurança da rede.

A AC está encarregue da emissão dos certificados dos diferentes nós que operam na rede. Como se encontra *offline*, estes certificados terão de ser emitidos antes do lançamento da rede ou através de outros métodos. A responsabilidade de aceitar determinado elemento num grupo, independentemente deste possuir um certificado, fica a cargo do líder desse grupo.

A possibilidade de receber os certificados ou certificados de revogação via outros métodos durante o funcionamento da rede, por exemplo, através de outros sistemas em funcionamento em escalões superiores, não foi contemplada neste estudo. Contudo, seria uma mais-valia para um melhor funcionamento do mesmo, sendo uma das propostas para trabalhos futuros a desenvolver nesta área. Se assim fosse, a AC já não seria considerada *offline*.

O modelo de certificação da AC também se encontra fora do estudo desta dissertação, uma vez que requer uma análise cuidada das diferentes áreas onde este sistema pode operar. Além disso, ao escolher-se um modelo de certificação específico estar-se-ia a limitar a adaptabilidade do sistema. O ideal seria escolher o modelo de certificação dependendo do contexto da missão ou da instituição em que a rede irá operar.

3.2.5 Tipo de Mensagem

De modo a não alterar qualquer um dos tipos de mensagens previstos originalmente no CRAN, é necessário criar um novo campo (Tipo Msg), isto para que seja possível aos nós discernirem entre os diferentes tipos de mensagens recebidas. Doravante, todos os formatos de mensagem apresentados irão conter este campo, sendo desnecessário explicá-lo nessas subsecções.

Este campo irá diferenciar 7 diferentes tipos de mensagens, sendo estas: as mensagens originais do CRAN, KA, *Join Request*, *Join Reply*, *Leave Request*, *Leave Reply* e *Key Refresh*. Tendo em conta os 7 tipos de mensagens, o tamanho deste campo será de 3 bits.

O formato destas mensagens obriga a que se saiba qual o seu tipo antes de ser efetuada a leitura dos seus campos, principalmente quando é necessário ler determinado conteúdo cifrado, pois é necessário saber qual o algoritmo e chave a utilizar para o decifrar.

3.2.6 *Nonce*

O *Nonce* é um campo que irá ser utilizado pelos tipos de mensagens que irão ser abordados ao longo deste capítulo. A sua função é, principalmente, evitar a replicação de mensagens e garantir que os nós que estão a comunicar, o façam sem que nenhum outro se faça passar por eles.

Este campo irá ser gerado com base na data e hora, permitindo assim aos nós recetores das mensagens, que estes comprovem que a mensagem recebida não foi replicada por um nó malicioso com o objetivo de destabilizar o funcionamento da rede.

3.2.7 Chaves

Os tipos de chaves projetadas de modo a garantir segurança neste sistema são apresentados de seguida:

- Chave de grupo/pelotão - Chave simétrica, partilhada entre nós do grupo/pelotão e utilizada para cifrar e decifrar todo o tipo de mensagens do protocolo CRAN transmitidas no seio do grupo/pelotão.
- Chave pública/privada – Cada nó tem associado a si um par de chaves assimétricas (pública e privada), utilizadas, principalmente, para pedidos de adesão e remoção de nós a/de um grupo.
- Chave de sessão - Chave simétrica, conhecida apenas pelo líder de um grupo e outro nó, de modo a que os processos de adesão e remoção de nós do grupo sejam mais rápidos, relativamente à utilização do par de chaves assimétricas. Este tipo de chave não foi utilizado no sistema proposto na presente dissertação, sendo remetido para trabalho futuro na subsecção 5.2.

3.2.8 Fila de potenciais líderes

Existe uma fila deste tipo presente em cada nó da rede. Esta fila contém os IDs dos nós, dos quais foi recebida informação de que estes são líderes de grupo. Os IDs dos potenciais líderes irão ser ordenados na fila por ordem hierárquica (patente), aos quais cada nó irá tentar fazer um pedido de adesão ao grupo desse líder. Note-se que, relativamente ao elemento em questão, apenas os IDs de elementos de patente superior à do seu atual líder são postos nesta fila.

3.3 *Keep Alive*

Este tipo de mensagem já foi explicado anteriormente em 2.1. Contudo, para a elaboração deste sistema de segurança, esta mensagem tem um papel importante. Sem alterar qualquer um dos campos originalmente estabelecidos para o formato desta mensagem, foram-lhe acrescentados dois campos:

- *Timestamp* (TS)- Este campo é preenchido com um índice temporal, que corresponde ao instante em que o nó líder de grupo gerou o seu KA;
- Certificado_Líder - Certificado do líder do grupo, ao qual o nó que gerou a mensagem pertence.

Na Figura 3.1 encontra-se um esquema do novo formato deste tipo de mensagem.

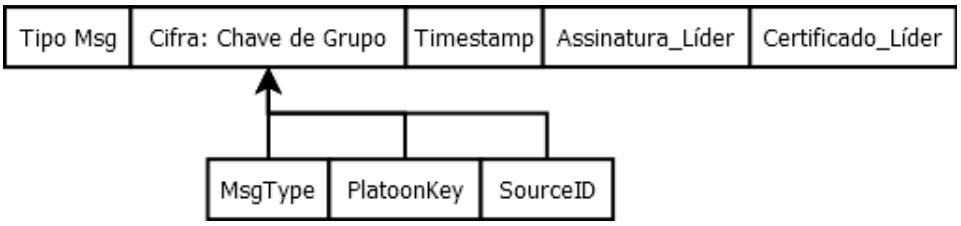

Figura 3.1: Formato da mensagem *Keep Alive*.

O *Timestamp* é necessário para anunciar aos outros nós a presença do líder no instante indicado no campo, isto para que os nós pertencentes à rede saibam que o líder ainda se encontra presente no grupo. A importância desta informação para o funcionamento da rede encontra-se explicada nas subsecções 3.7.3 e 3.7.4. Este campo é assinado pelo nó líder, garantindo assim a integridade e autenticidade da informação presente. Os nós que queiram emitir este campo nos seus KAs, apenas precisam de o copiar de outro KA recebido, uma vez que não conseguem assiná-lo pelo líder.

O certificado do líder tem de ser enviado na mensagem por duas razões. É necessário que os nós recetores tenham acesso à chave pública do líder para comprovar a validade do *Timestamp*. A outra razão prende-se com o facto de que, caso um nó externo à rede queira fazer um pedido de adesão, este irá precisar do ID do líder, como irá ser explicado posteriormente na subsecção 3.4.1.

Estes dois novos campos adicionados fazem com que esta mensagem, que inicialmente foi projetada para anunciar a presença do nó que a emite, agora sirva também o propósito de propagar o sinal do líder de grupo (*Beacon*) por todos os nós da rede e externos que se queiram juntar à mesma.

Um nó que faça parte do grupo e receba esta mensagem, verifica se:

- O certificado é o do seu líder;
- A assinatura do *Timestamp* corresponde à do seu líder;
- O *Timestamp* não expirou.

Caso estas condições se verifiquem, o *Timestamp* é renovado.

O restante da mensagem contém os campos originais do KA previstos no CRAN. Basta decifrar com a chave de grupo para ter acesso a esses campos.

Um nó que não faça parte do grupo e receba esta mensagem, verifica se:

- O certificado é válido;
- A assinatura do *Timestamp* corresponde à do nó cujo ID se encontra presente no certificado;
- O *Timestamp* não expirou;

- A patente presente no certificado é mais alta que a do seu atual líder.

Caso estas condições se verifiquem, o nó coloca o ID do certificado na fila de potenciais líderes (3.2.8).

3.4 Adição de Nós

Esta operação tem como objetivo resolver o problema de adesão de nós ao grupo. Estes nós apenas podem aderir ao grupo mediante autorização do líder.

3.4.1 *Join Request*

O formato deste tipo de mensagem consiste em 5 campos:

- Tipo Msg – Tipo da mensagem, neste caso, *Join Request*;
- ID_Novo_Líder – Identificação do nó a quem é feito o pedido de adesão ao grupo;
- *Nonce* – Utilizado de modo a garantir autenticação entre o nó que faz o pedido e o líder do grupo e evita ainda a replicação das mensagens;
- Assinatura_Nó_Ext – Assinatura digital dos campos ID_Novo_Líder e *Nonce*, do nó que faz o pedido de adesão;
- Certificado_Nó_Ext - Certificado do nó que faz o pedido de adesão ao grupo.

Na Figura 3.2, encontra-se um esquema do formato deste tipo de mensagem.

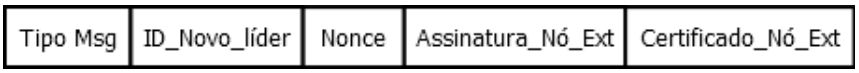

Figura 3.2: Formato da mensagem *Join Request*.

Os campos ID_Novo_Líder e *Nonce* são assinados pelo nó que faz o pedido de adesão ao grupo. Esta assinatura irá impossibilitar que um nó malicioso que pretenda modificar a mensagem, o consiga fazer, pois apenas o nó que a gerou a pode ter assinado.

Devido ao tamanho dos certificados ser considerável em relação ao resto dos campos e, tendo em consideração a possibilidade da rede operar com tecnologias de reduzida largura de banda, o funcionamento do sistema pode ser comprometido. Uma solução pode passar por estabelecer *a priori* os certificados que irão ser usados em cada nó. Deste modo, evitar-se-ia o envio dos certificados, tanto neste tipo de mensagens como nas do tipo *Keep Alive* (subsecção 3.3), bastando enviar apenas o ID do nó emissor, para que seja possível associá-lo ao seu certificado após a receção da mensagem. Como é expectável, em situações em que esta solução seja adotada é preciso que se saiba de antemão quais os nós que irão ser utilizados, caso contrário, novos elementos cuja participação na rede não foi prevista inicialmente, terão de enviar o seu certificado.

Qualquer nó intermédio que receba a mensagem, ao ler o tipo de mensagem (*Join Request*), sabe que esta tem de ser reencaminhada para o nó do líder. Caso o ID_Novo_Líder não seja o mesmo que o ID do líder do nó intermédio, a mensagem não é reencaminhada. Se um dos nós que recebe esta mensagem não tiver mais do que um nó vizinho, a mensagem também não é reencaminhada, pois não existe nenhum outro vizinho para receber a mensagem, exceto aquele que a enviou anteriormente.

Ainda na sequência do paragrafo anterior, existem duas soluções para como é tomada a decisão de reencaminhar ou não uma mensagem. Existe a possibilidade de cada nó intermédio verificar a integridade da mensagem recebida antes de a reencaminhar ou de deixar essa verificação a cargo do nó de destino. Na tabela 3.1 são apresentadas as vantagens e desvantagens para a solução, na qual a integridade da mensagem é verificada nos nós intermédios, sendo que as vantagens desta solução, são as desvantagens da outra e *vice-versa*.

Tabela 3.1: Verificação da integridade das mensagens em nós intermédios: vantagens e desvantagens.

Vantagens	Desvantagens
Assim que o primeiro nó receba uma mensagem cuja integridade falhe, evita-se o reenvio da mesma por todos os elementos da rede.	Cada nó que receber uma mensagem, irá verificar a sua integridade. Dependendo do tempo que seja necessário para processar a validação de cada uma destas assinaturas, pode significar um grande tempo de atraso.
Evita que o inimigo consiga perturbar o funcionamento da rede através da continua emissão de mensagens deste tipo, uma vez que estas são logo descartadas nos primeiros elementos.	

O campo *Nonce* tem um critério temporal e existe, pois é necessário garantir duas condições importantes. É preciso garantir que a receção da mensagem seja feita dentro de uma determinada janela temporal, de modo a evitar a replicação de mensagens. Outra das condições é garantir ao nó aderente, no momento em que este recebe a resposta, que o seu pedido foi recebido e tratado pelo líder, sendo este *Nonce* enviado na resposta de volta para o nó aderente.

O certificado é enviado em conjunto com a mensagem por três razões:

- É necessário que o nó líder tenha acesso ao certificado do nó aderente, para poder confirmar a sua autenticidade e a sua validade perante a autoridade certificadora;
- É necessário que o nó líder tenha acesso à chave pública do nó aderente para confirmar a assinatura da mensagem e posteriormente enviar a chave de grupo cifrada pela primeira, caso o pedido seja aceite;
- É necessário que o nó líder tenha acesso ao ID do nó aderente, para poder enviar a resposta. Esta solução é mais atrativa do que uma em que seja necessário guardar o caminho feito pelo

Join Request na própria mensagem (*Source Routing*), uma vez que o certificado já terá de ser enviado pelas razões acima referidas, não aumentando o tamanho da mensagem por cada salto entre nós que seja efetuado. Além disso, como se trata de uma MANET, a probabilidade de alteração de todo o percurso realizado pela mensagem é elevada, levando à sua possível perda, caso se optasse pelo *Source Routing*.

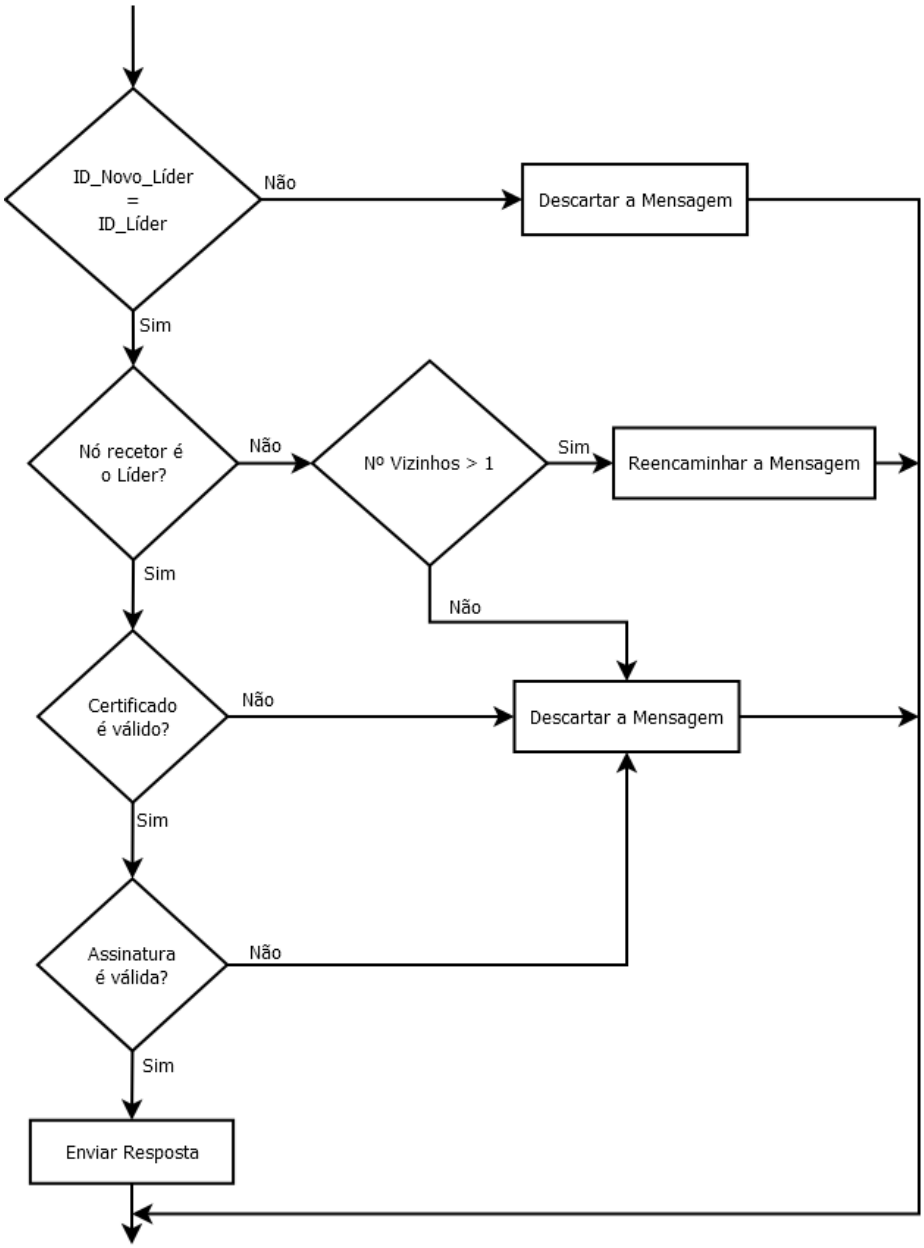

Figura 3.3: Gestão da receção de um *Join Request*.

O nó líder, após receção da mensagem, verifica se:

- O ID na mensagem é o seu, confirmando assim que é o recetor da mensagem;
- É líder de grupo neste momento, pois não faz sentido receber uma mensagem de do tipo *Join Request* caso não seja efetivamente o líder de um grupo;

- O certificado é válido, caso contrário este nó não poderá aderir à rede;
- A assinatura é verificável com a chave pública do certificado, pois é necessário verificar a integridade e autenticidade da mensagem.

Caso todas estas condições se verifiquem e o líder aceite que esse nó faça parte do grupo, então é enviada uma resposta com sucesso e com a chave de grupo/pelotão. Caso contrário, é enviado uma resposta sem sucesso.

O certificado do nó aderente deverá ser guardado pelo líder para futura utilização, esta utilização irá ser explicada nas próximas secções.

Na Figura 3.3 encontra-se representado um fluxograma representativo de como é gerida a receção de uma mensagem deste tipo.

3.4.2 *Join Reply*

O formato deste tipo de mensagem consiste em 6 campos:

- Tipo Msg – Tipo da mensagem, neste caso, *Join Reply*;
- *Nonce* – Cópia do *Nonce* enviado pelo nó externo no pedido de adesão;
- Chave de Grupo/Pelotão – Caso o líder de grupo tenha aceite o pedido de adesão, este campo é preenchido pela chave de grupo;
- ID_Nó_Ext - Identificação do nó que fez o pedido de adesão ao grupo;
- Resposta - Sucesso, caso o pedido de adesão seja aceite. Insucesso, caso contrário;
- Assinatura_Líder - Assinatura digital do nó líder.

Na Figura 3.4 encontra-se um esquema do formato deste tipo de mensagem.

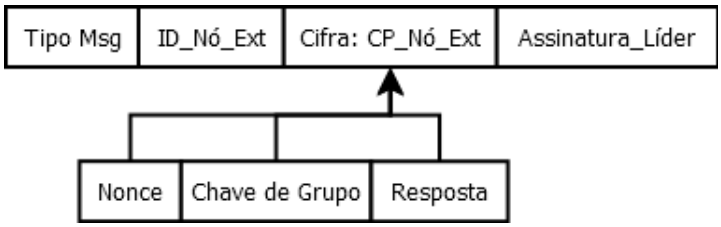

Figura 3.4: Formato da mensagem *Join Reply*.

Os campos *Nonce*, Chave de Grupo/Pelotão e Resposta são cifrados pela chave pública do nó que fez o pedido de adesão. Esta irá impedir que um nó malicioso consiga obter a chave de Grupo/Pelotão, uma vez que para o fazer seria necessário possuir a chave privada do nó aderente. Posteriormente, esta cifra e ainda o campo ID_Nó_Ext, são assinados pelo líder. Um nó malicioso que pretenda modificar a mensagem não o consegue fazer, uma vez que esta apenas poderá ter sido assinada pelo nó líder.

Qualquer nó intermédio que receba esta mensagem, ao ler o seu tipo (*Join Reply*), sabe que esta tem de ser reencaminhada para o nó cujo ID está presente no campo ID_Nó_Ext. O processo de

reencaminhamento desta mensagem encontra-se de acordo com os critérios definidos anteriormente para as mensagens do tipo *Join Request* na subsecção 3.4.1.

A necessidade do campo *Nonce* encontra-se igualmente explicada na subsecção 3.4.1.

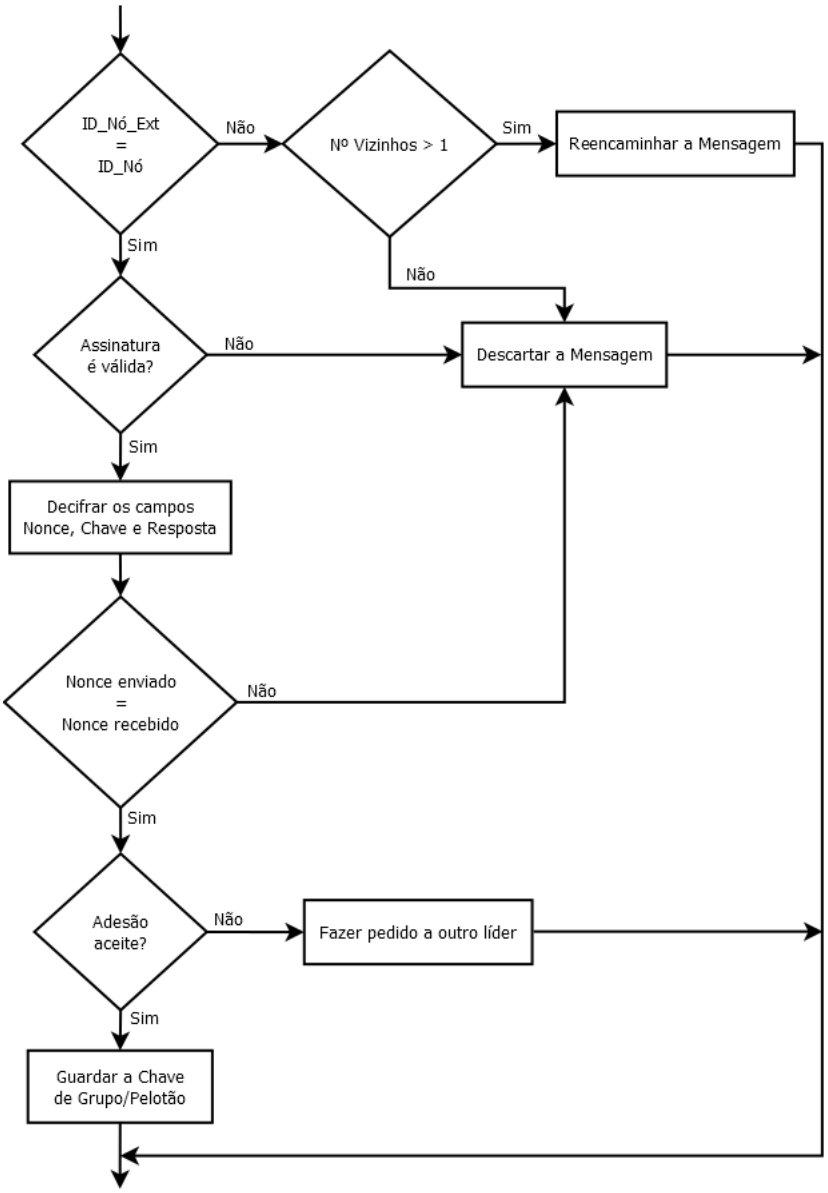

Figura 3.5: Gestão da receção de um *Join Reply*.

O nó aderente, após a receção da mensagem, verifica se:

- ID_Nó_Ext é o seu, garantindo assim que é o destinatário da mensagem;
- A assinatura é verificável com a chave pública do nó líder, pois é necessário verificar a integridade e autenticidade da mensagem.
- É possível decifrar os campos *Nonce*, Chave de Grupo/Pelotão e Resposta;
- A resposta não é Insucesso, caso o seja, não poderá aderir à rede;
- O *Nonce* enviado e recebido são os mesmos.

Caso todas estas condições se verifiquem, o nó aderente passa a ter acesso à chave de grupo, fazendo parte da rede a partir desse momento e podendo enviar/receber para/de os restantes nós da rede de forma confidencial as mensagens previstas no CRAN.

Na Figura 3.5 encontra-se representado um fluxograma representativo de como é gerida a receção de uma mensagem deste tipo.

3.4.3 Diagrama temporal

Após a explicação dos diferentes tipos de mensagens associados ao processo de adesão à rede, torna-se importante, para uma melhor compreensão, a apresentação de um diagrama temporal que ilustra a troca de mensagens entre nós relativas ao processo de adesão ao grupo.

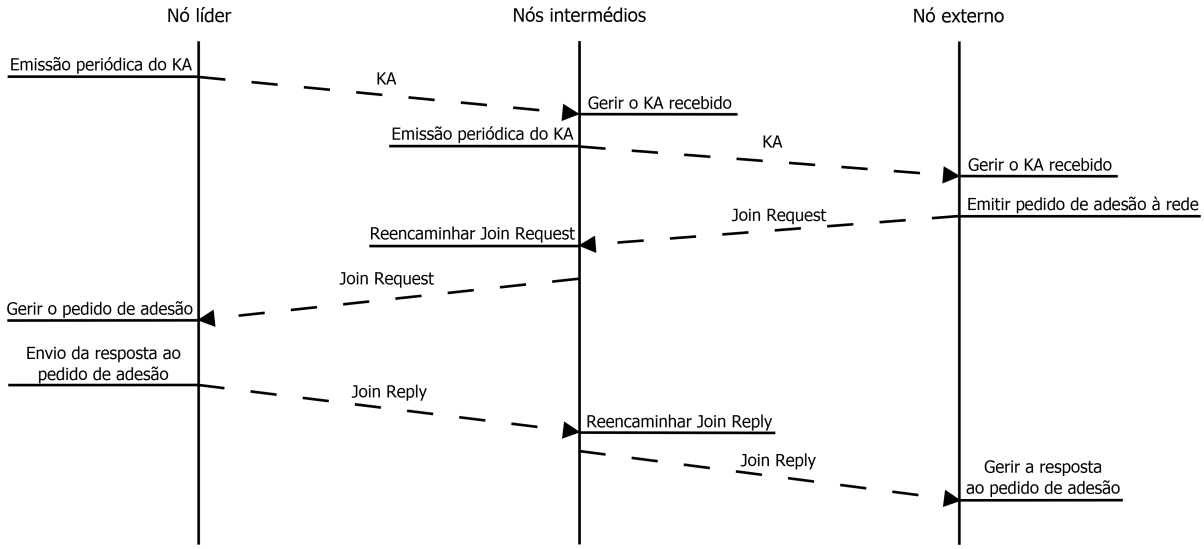

Figura 3.6: Diagrama temporal do processo de adesão de um nó ao grupo.

A gestão que é feita pelos nós após a receção do KA, *Join Request* e *Join Reply*, encontra-se explicada nas Figuras 3.13, 3.3 e 3.5 nas subsecções 3.7.3, 3.4.1 e 3.4.2, respetivamente.

3.5 Remoção de Nós

Este processo permite aos nós pertencentes ao grupo, pedir autorização ao líder para sair do mesmo. Esta capacidade do sistema revela-se importante, não só pelo facto de dar conhecimento ao líder do grupo da alteração que poderá ocorrer caso o pedido seja aceite, mas também por alertá-lo para a necessidade de gerar uma nova da chave de grupo.

O modo como foi estruturado este processo assemelha-se em grande parte ao processo de Adição de Nós presente na subsecção 3.4, sendo para este efeito usados 2 tipos de mensagens, *Leave Request* e *Leave Reply*.

3.5.1 *Leave Request*

O formato deste tipo de mensagem é idêntico ao do tipo *Join Request* (subsecção 3.4.1). Fazendo uma analogia entre os tipos de mensagem *Join Request* e *Leave Request*, é possível entender facilmente as funções destes campos no segundo. O formato desta mensagem consiste portanto, em 5 campos:

- Tipo Msg – Tipo da mensagem, neste caso, *Leave Request*;
- ID_Líder – Identificação do nó líder, a quem é feito o pedido de saída do grupo;
- *Nonce* – Utilizado de modo a garantir autenticação entre o nó que faz o pedido e o líder do grupo, evita ainda a replicação das mensagens;
- Assinatura_Nó_Saída – Assinatura digital dos campos ID_Líder e *Nonce*, do nó que faz o pedido de saída;
- Certificado_Nó_Saída - Certificado do nó que faz o pedido de saída do grupo.

Na Figura 3.7 é possível observar um esquema do formato deste tipo de mensagem, bem como a analogia feita entre este e o tipo *Join Request*.

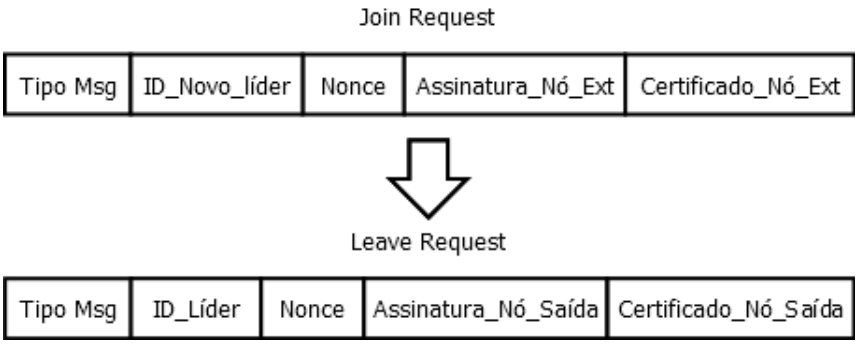

Figura 3.7: Analogia entre os formatos *Join Request* e *Leave Request*.

Os campos ID_Líder e *Nonce* são assinados pelo nó que faz o pedido de saída, garantindo assim a integridade da mensagem.

Os nós intermédios pertencentes ao grupo procedem ao reencaminhamento das mensagens de acordo com os critérios definidos anteriormente (subsecção 3.4.1).

O nó líder, após receção da mensagem, executa os mesmo procedimentos de verificação explicados em 3.4.1.

Posteriormente, caso o pedido de saída seja aceite, é necessário efetuar a renovação da chave de grupo. Para tal, segue-se o mesmo processo que se encontra explicado na subsecção 3.6.

3.5.2 *Leave Reply*

O formato deste tipo de mensagem já apresenta algumas alterações em relação ao tipo *Join Reply*, isto porque deixa de ser necessária a partilha da chave de grupo. Assim sendo, o formato desta mensagem consiste em 5 campos:

- Tipo Msg – Tipo da mensagem, neste caso, *Leave Reply*;
- ID_Nó_Saída – Identificação do nó que fez o pedido de saída ao grupo;
- *Nonce* – Cópia do *Nonce* enviado pelo nó que fez o pedido de saída;
- Resposta - Sucesso, caso o pedido de saída seja aceite. Insucesso, caso contrário;
- Assinatura_Líder – Assinatura digital do nó líder.

Na Figura 3.8 encontra-se um esquema do formato deste tipo de mensagem:

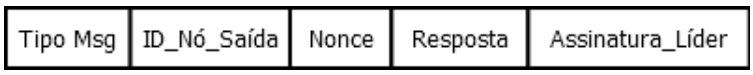

Figura 3.8: Formato da mensagem *Leave Reply*.

Os campos ID_Nó_Saída, *Nonce* e Resposta são assinados pelo líder, por forma a garantir a integridade da mensagem.

Uma vez mais, o procedimento para o reencaminhamento das mensagens encontra-se de acordo com os critérios definidos em 3.4.1.

O nó que realizou o pedido de saída, ao receber esta mensagem, verifica se:

- ID_Nó_Saída é o seu, garantindo assim que é o destinatário da mensagem;
- A assinatura é verificável com a chave pública do nó líder, pois é necessário verificar a integridade e autenticidade da mensagem.
- A resposta não é Insucesso, caso o seja, não tem autorização para sair da rede;
- O *Nonce* enviado e recebido são os mesmos.

Caso estas condições se verifiquem, então o nó está autorizado a sair do grupo. A chave de grupo que este possuí tornar-se-à obsoleta, uma vez que será eleita uma nova chave de grupo pelo líder.

3.6 Expulsão de Nós

Para resolver o problema de remoção de nós do grupo, é necessário garantir que os nós que serão removidos, não consigam aceder às novas mensagens de dados partilhadas entre o grupo. Assim sendo, terá de ser gerada uma nova chave de grupo por parte do seu líder e difundida pelos nós que permanecem na rede, sem que os removidos consigam obtê-la. Para tal, é proposto um outro tipo de mensagem, *Key Refresh*.

O formato deste tipo de mensagem consiste em 5 campos:

- Tipo Msg – Tipo da mensagem, neste caso, *Key Refresh*;
- ID Dest – Identificador do nó recetor da mensagem, que será um dos nós que permanece no grupo;

- Chave de Grupo/Pelotão – Este campo é preenchido com a nova chave de grupo/pelotão;
- *Nonce* - Utilizado de modo a evitar a replicação da mensagem.
- Assinatura_Líder - Assinatura digital do nó líder.

Na Figura 3.9 encontra-se um esquema do formato deste tipo de mensagem.

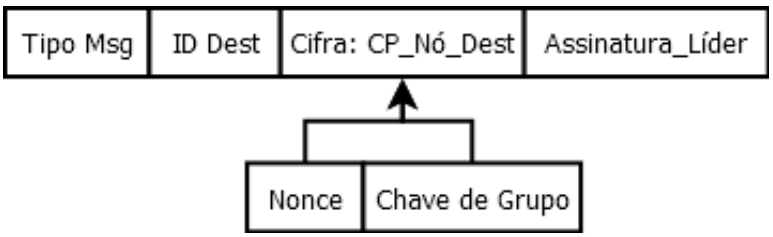

Figura 3.9: Formato da mensagem *Key Refresh*.

Os campos *Nonce* e Chave de Grupo são cifrados pela chave pública do nó que irá receber a mensagem, impedindo que qualquer um dos nós que vão ser removidos consiga obter algum destes campos. Posteriormente, esta cifra e ainda o campo ID Dest, são assinados pelo líder, garantindo assim a integridade e autenticidade da mensagem.

Uma vez mais, o procedimento de um nó intermédio que receba esta mensagem está de acordo com os critérios definidos anteriormente para as mensagens do tipo *Join Request* e *Join Reply* na subsecção 3.4.1.

O número de mensagens deste tipo, a serem enviadas, será igual ao número de nós que permanecem na rede. É gerada uma mensagem para cada um destes nós. Na Figura 3.10 é apresentado um diagrama temporal que ilustra o processo de remoção de nós de um grupo.

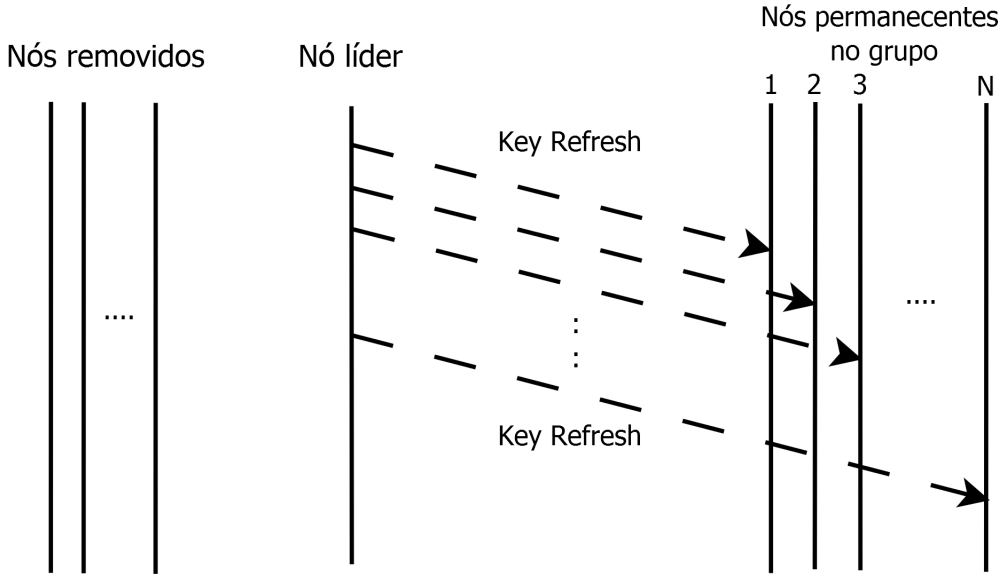

Figura 3.10: Diagrama temporal do processo de remoção de nós do grupo.

O nó recetor, verifica se:

- O ID Dest é o seu, garantindo assim que é o destinatário da mensagem;

- A assinatura é do seu líder de grupo, utilizando para isso a chave pública do nó líder de grupo, verificando a integridade e autenticidade da mensagem;

- O *Nonce* não é igual a um outro utilizado anteriormente, prevenindo assim a replicação da mensagem.

Na Figura 3.11 encontra-se representado um fluxograma representativo de como é gerida a receção de uma mensagem deste tipo, que acaba por ser uma versão mais simples da receção de uma mensagem do tipo *Join Reply*.

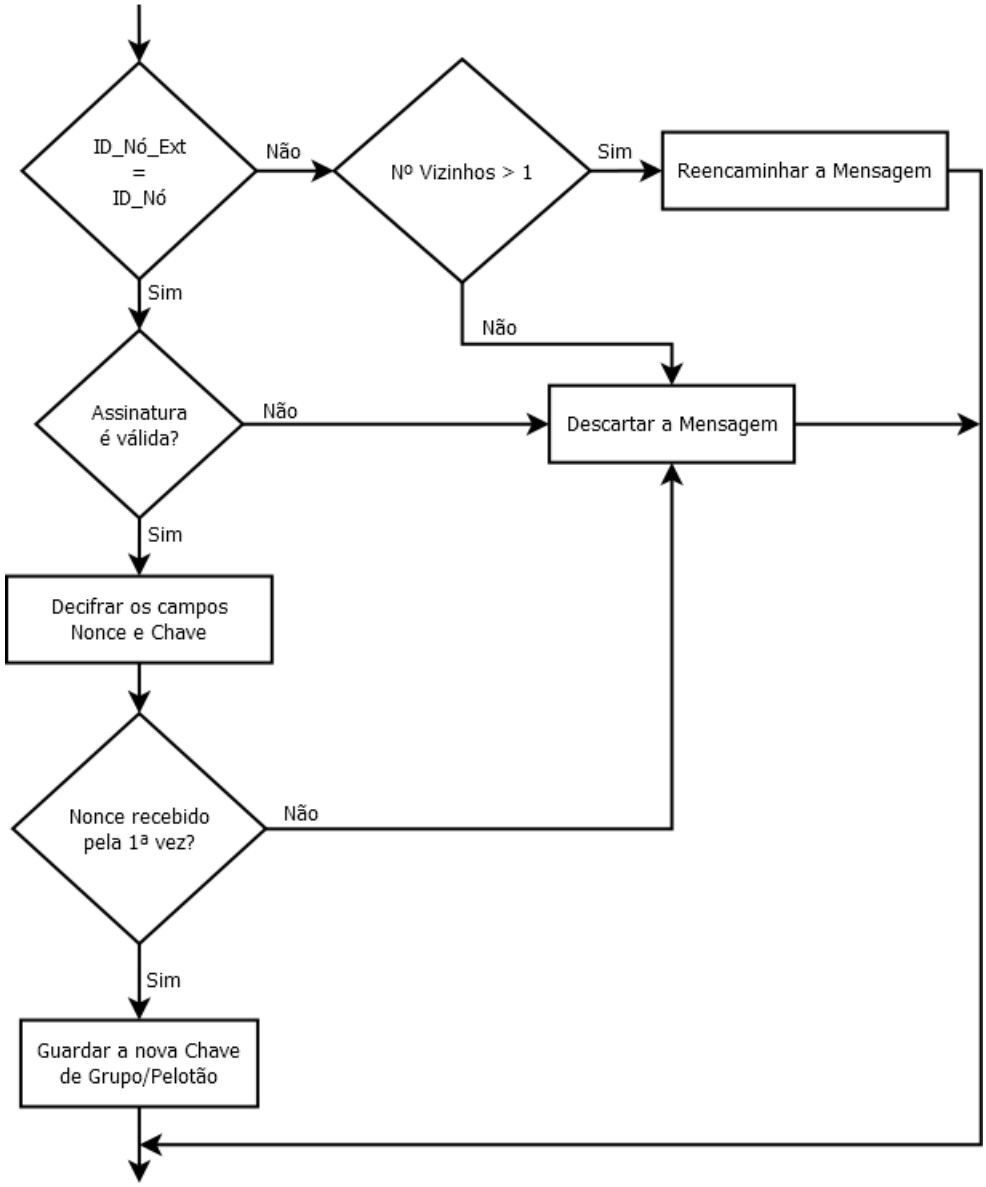

Figura 3.11: Gestão da receção de um *Key Refresh*.

3.7 Outros aspetos relevantes

Nesta subsecção são abordados aspetos críticos para o funcionamento deste sistema, cuja reflexão é necessária por forma a evitar problemas na rede. Assim sendo, são apresentados os potenciais desafios, bem como as suas soluções.

3.7.1 Processo de reencaminhamento de mensagens

Para garantir que as mensagens dos tipos, *Join Request*, *Join Reply*, *Leave Request*, *Leave Reply* e *Key Refresh* não sejam reencaminhadas infinitamente, cada nó que as reencaminhe de acordo com as regras definidas na subsecção 3.4.1, apenas o faz uma vez. Para tal, cada mensagem de qualquer destes tipos que seja recebida, é guardada em memória ao longo de toda a operação da rede. Posteriormente, ao receber outras mensagens deste tipo, estas são comparadas com as recebidas anteriormente. Deste modo, é possível saber se é ou não a primeira vez que a dita mensagem está a ser recebida, podendo ser reencaminhada desde que se encontre de acordo com os outros critérios definidos. A Figura 3.12 ilustra o funcionamento deste processo.

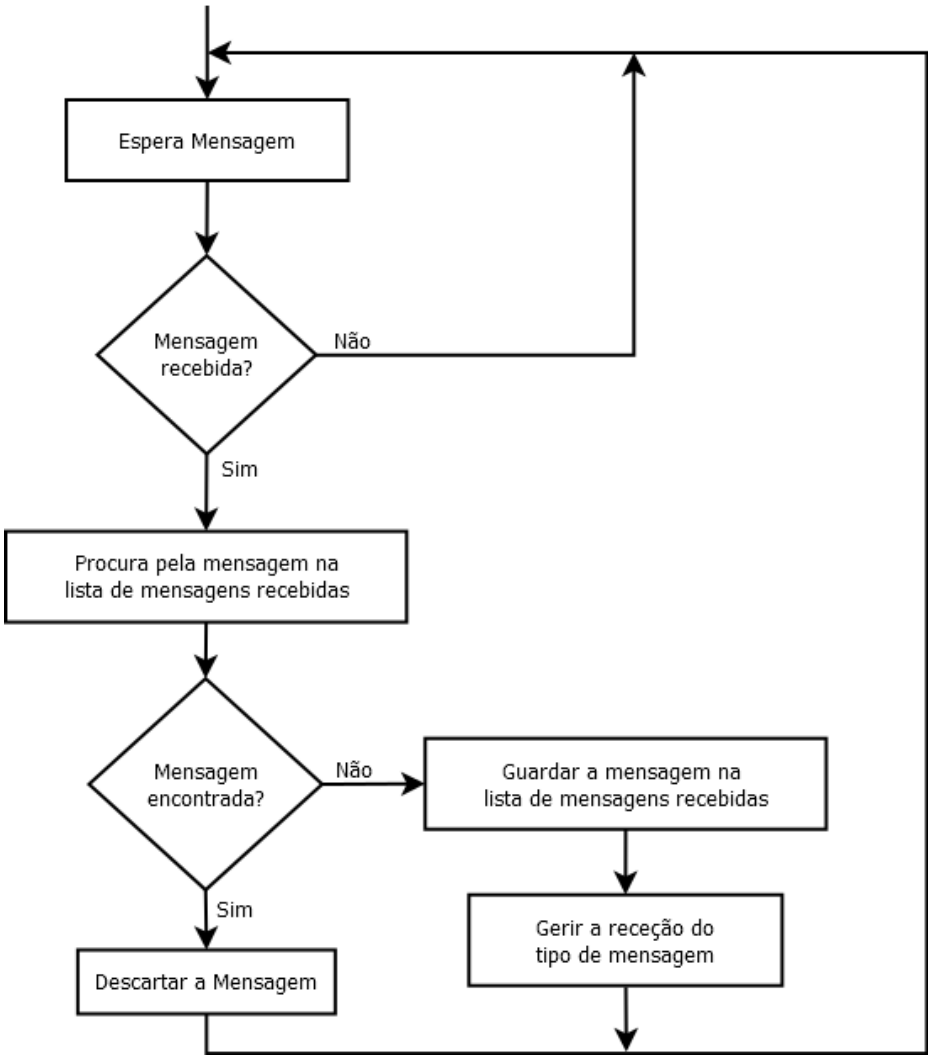

Figura 3.12: Processo de reencaminhamento das mensagens.

3.7.2 Modo *PFS*

Uma vez que pode existir a necessidade de garantir que mensagens trocadas entre um grupo, não sejam decifradas por nós que adiram ao mesmo *a posteriori* (daí o nome atribuído), a existência deste modo torna-se fulcral para o sistema. Para tal, sempre que um nó subalterno adere a um grupo, a chave de grupo tem de ser renovada e distribuída pelos restantes nós. Isto consegue-se utilizando o tipo de mensagem *Key Refresh*, para distribuir a nova chave de grupo por todos os nós subalternos, assim que o pedido de adesão seja aceite.

Este modo, devido à necessidade da distribuição da nova chave de grupo, requer um maior tráfego de mensagens na rede, face ao modo em que a chave não é renovada. Assim sendo, a utilização deste modo pode ser decidida antes ou durante o funcionamento da rede, decisão esta que recai uma vez mais sobre o compromisso entre a segurança e a operacionalidade da rede.

3.7.3 Deteção da partição

No que toca à segurança do CRAN, torna-se necessário que sejam detetadas eventuais partições de um pelotão em 2 ou mais subgrupos (quando não existe ligação possível com outro/s nó/s). Esta característica não estava prevista inicialmente no CRAN uma vez que não existiam chaves de pelotão para cifrar as mensagens, de modo que a partição não era detetada e nem sequer alterava o funcionamento da rede.

A solução para este problema passa por utilizar um dos tipos de mensagens previstos no CRAN, as mensagens de *Keep Alive* (KA). Sempre que um nó envia uma mensagem deste tipo, acrescenta-lhe ainda dois campos, um com um índice temporal (Timestamp) assinado pelo seu líder e o outro com o certificado deste. Sempre que um nó receba uma mensagem de KA, este procederá da forma seguinte:

- Verificar se o ID presente no certificado corresponde ao seu atual líder (para trabalho futuro poder-se-á verificar apenas se o certificado é igual ao do seu líder atual, não sendo necessário comprovar a validade do certificado, pois isso já foi feito anteriormente);

- Verificar se a patente presente no certificado é mais alta que a do seu atual líder, colocando o ID numa fila por ordem de patente;

- Verificar se o *Timestamp* ainda não expirou (para tal, é necessário utilizar a chave pública do nó líder, presente no certificado).

Para um melhor esclarecimento, apresenta-se na Figura 3.13 um fluxograma que ilustra este processo. Deste modo é possível a qualquer nó presente na rede detetar se se encontra numa partição. Quando é detetada uma partição do pelotão, o subgrupo que ficou sem o líder necessita que um seja eleito, para que possa continuar a ser feita a gestão do grupo.

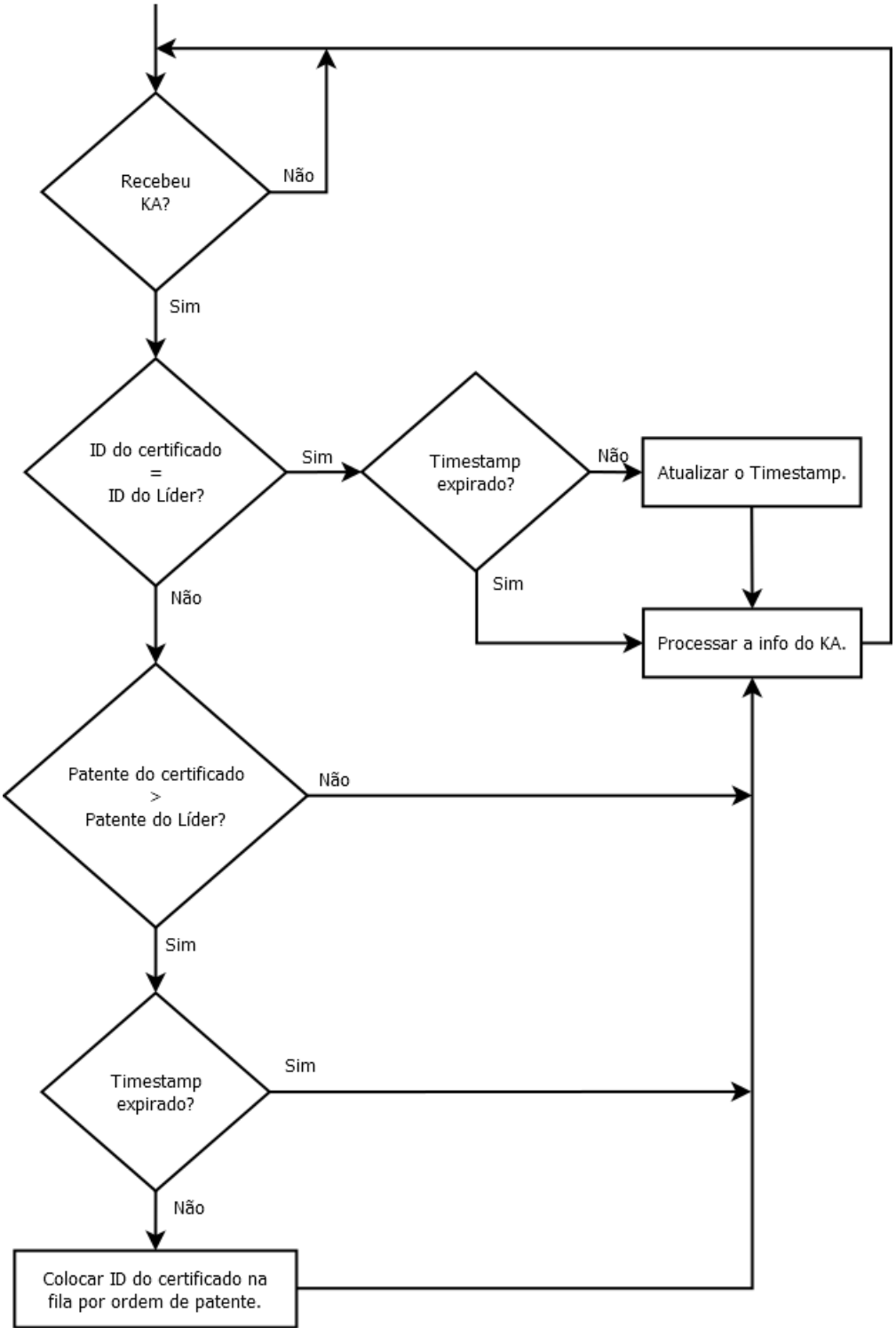

Figura 3.13: Fluxograma do processo de deteção de uma partição.

3.7.4 Eleição de líder de grupo

Para a eleição do líder de um grupo, é proposta uma solução que faz uso da fila de potenciais líderes (subsecção 3.2.8) existente em cada nó. O modo como a fila referida é construída encontra-se explicado na subsecção 3.7.3 e é parte integrante do fluxograma ilustrado na Figura 3.13.

Nesta solução, cada nó retira o ID do nó que se encontra na frente da fila e executa um pedido de

adesão a esse nó. Caso o pedido seja aceite, então o nó que fez o pedido de adesão muda o seu líder e limpa a fila. Caso o pedido seja rejeitado, então volta-se a retirar um ID da frente da fila e recomeça-se o processo. Este processo é executado sempre que a fila não se encontre vazia. Caso a fila esteja vazia, então o nó assume-se como líder do grupo, passando a acrescentar o seu certificado e o *Timestamp* assinado por si à mensagem de KA.

De seguida apresenta-se um fluxograma explicando o funcionamento deste processo:

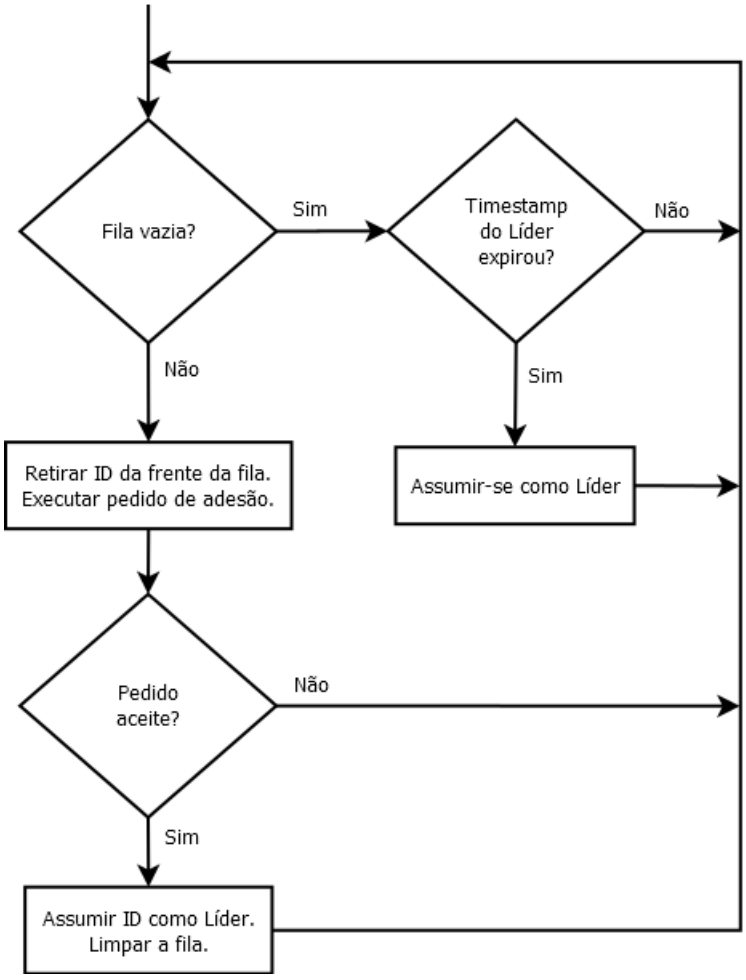

Figura 3.14: Fluxograma do processo de eleição de líder.

3.7.5 *Rejoin*

Para que dois ou mais grupos, que se tenham separado anteriormente do seu pelotão, se possam voltar a juntar, é necessário que consigam detetar quando estão ao alcance um do outro. A solução para este problema passa por executar o processo de eleição de líder de grupo (subsecção 3.7.4), no entanto, esse processo é desnecessário caso a chave de grupo utilizada por ambos os grupos ainda seja a mesma que estava em vigor no momento da partição, facilitando a junção dos grupos (os nós podem assumir o líder com patente mais alta automaticamente, sem ter de realizar um pedido de adesão). Para saber se a chave de grupo em utilização é a mesma, basta decifrar a mensagem de KA com essa chave e verificar a sua integridade.

3.8 Algoritmos criptográficos

Nesta subsecção, são definidos os algoritmos criptográficos utilizados neste sistema, bem como a justificação para a sua escolha. Note-se que, dependendo do tipo de ambiente em que esta rede opera e da sua duração, outros algoritmos poderão revelar-se mais viáveis, desde que nunca prejudiquem a segurança da mesma. Por exemplo, caso a rede esteja prevista atuar num cenário de longa duração, chaves de maior tamanho poderão ser necessárias ou até mesmo a alteração dos algoritmos utilizados. Por outro lado, chaves de tamanho mais reduzido podem ser usadas em cenários de curta duração, levando a um menor tamanho do *payload* das mensagens.

3.8.1 Cifra simétrica

O algoritmo de cifra escolhido, sob o qual correm todas as mensagens do protocolo CRAN, é o AES. Este algoritmo foi eleito em detrimento do DES, pois no que diz respeito à segurança, uma chave DES é considerada fraca para os dias de hoje. Além disso, a vulnerabilidade do DES relativamente a CAD e CAL, tornam o algoritmo ainda menos seguro.

Dentro das 3 opções para o AES (128, 194 e 256 bits), foi escolhida a opção de 128 bits por duas razões. A primeira prende-se com o facto de que, de entre as 3 opções, esta é a opção na qual cada bloco de cifra é processado mais rapidamente. Mesmo tendo em consideração que as outras opções poderão ser mais seguras, como já foi referido anteriormente, é necessário fazer um compromisso entre o tamanho da cifra e a segurança que esta confere. Uma vez que 128 bits é suficiente para um funcionamento seguro do sistema, não há razão para escolher as opções de 194 ou 256 bits. Esta justificação dá azo à segunda razão pela qual o AES-128 foi o algoritmo escolhido. Tendo em conta que grande parte do tráfego de mensagens do protocolo CRAN é de tamanho menor que 128 bits, caso se escolhesse uma das outras opções, estar-se-ia a utilizar um tamanho de bloco maior para transmitir a mesma quantidade de informação. Isto levaria à utilização de maior largura de banda, que é algo reduzido nos cenários onde este sistema irá operar.

3.8.2 Cifra assimétrica

O algoritmo de cifra assimétrica escolhido é o *ECC*, isto porque, de acordo com [5], os parâmetros utilizados para este algoritmo são menores quando comparados com algoritmos que não sejam de curvas elípticas. Assim é possível obter chaves e cifras mais curtas para o mesmo nível de segurança.

Uma vez que existem curvas recomendadas pelo *National Institute of Standards and Technology* (*NIST*) com parâmetros já definidos, isto permite que ao escolher uma delas para ser utilizada em todo o sistema, se reduza o tamanho do certificado, sendo apenas necessário que este contenha o parâmetro correspondente à chave pública, no que diz respeito a este algoritmo. Por consequência, o *payload* das mensagens nas quais os certificados são enviados, serão menores.

3.8.3 Assinatura Digital

A escolha do algoritmo de assinatura digital baseou-se em [19], visto que, por razões relacionadas com a redução do *payload* nas mensagens, deve ser escolhido um algoritmo de curvas elípticas, cujas chaves, para uma segurança equivalente, são menores em comparação com outros algoritmos de assinatura. Posto isto, o algoritmo escolhido foi o *Elliptic Curve Digital Signature Algorithm* (*ECDSA*).

A ideia apresentada no segundo parágrafo da subsecção anterior (3.8.2), teve também influência na escolha do algoritmo de assinatura.

Reitera-se que outros algoritmos poderão ser utilizados caso as necessidades de segurança ou largura de banda assim o exijam ou permitam. Não apenas no que diz respeito à assinatura digital, mas também em relação às cifras simétricas e assimétricas (3.8.1 e 3.8.2, respetivamente).

3.9 Tempos de atraso

Neste sistema de segurança existem dois fatores importantes a considerar no que diz respeito aos tempos de atraso: o tempo necessário para a emissão das mensagens e o tempo de processamento.

O tempo necessário para a emissão de uma mensagem está diretamente ligado ao tamanho da mesma e ao ritmo binário com que esta consegue ser enviada. O tamanho das mensagens depende única e exclusivamente dos campos destas, daí que, durante o projeto do sistema, tentou-se sempre reduzir ao máximo o tamanho necessário para cada um destes, com especial foco nas cifras a serem utilizadas, sem prejudicar a segurança do sistema. Outro dos aspetos considerados no tamanho dos campos, foi a utilização de apenas o número de bits necessário à transmissão da informação pretendida, evitando a emissão desnecessária de bits e, consequentemente, uma menor eficiência. Em relação ao ritmo binário, este está dependente apenas da capacidade do sistema rádio a ser utilizado. Mais uma vez, reitera-se que se pretende que este sistema funcione num ambiente no qual o ritmo binário disponível restringe as decisões de implementação.

No que diz respeito ao tempo de processamento, uma vez que são utilizados algoritmos de cifra e assinaturas digitais com bastante frequência, todo este processo pode tornar-se moroso. A importância deste tempo no cálculo dos tempos de atraso do sistema, leva a que sejam consideradas algumas opções no que toca à otimização deste processo. Assim sendo, é possível aproveitar os momentos em que não esteja a ser usado todo o poder de processamento do nó para preparar com antecedência uma eventual assinatura ou cifra que será garantidamente utilizada. Este aproveitamento está dependente da capacidade de processamento do nó e do intervalo de tempo em que este se encontra livre para o fazer. No melhor caso, é possível fazer com que o tempo de atraso, entre o momento em que estas mensagens estejam agendadas e o tempo em que elas são efetivamente emitidas, seja zero. No pior caso, o tempo de atraso será igual ao tempo de processamento necessário para gerar as cifras ou assinaturas digitais pretendidas.

Abaixo, são apresentadas as mensagens onde se pode tirar partido desta otimização:

- *Keep Alive* – Uma vez que este tipo de mensagem é repetido periodicamente e todos os seus

campos podem ser preparados e cifrados/assinados previamente, é possível ter a mensagem totalmente preparada no momento em que esta está agendada para ser enviada;

- *Join Request* – Esta mensagem necessita apenas da chave privada do próprio nó para que seja possível assinar os seus campos. Assim sendo, embora a mensagem só possa ser gerada a partir do momento em que se saiba qual o potencial líder, é possível que enquanto se espera pela resposta deste, se gere mais mensagens deste tipo para outros potenciais líderes. Isto precavendo a possibilidade do primeiro pedido de adesão realizado seja rejeitado;

- *Key Refresh* – O nó líder pode preparar todas as mensagens deste tipo para serem enviadas para os nós do grupo, previamente. Para o fazer, em primeiro lugar, terá de gerar uma nova chave de grupo/pelotão, de seguida terá de cifrar os campos respetivos a este tipo de mensagem com a chave pública dos nós, e por fim assina a mensagem. Todos estes passos podem ser realizados com antecedência, fazendo com que as mensagens já estejam prontas a ser enviadas no momento de renovação da chave de grupo/pelotão.

A escolha dos algoritmos de cifra e assinatura digital a ser utilizados neste sistema basearam-se num compromisso entre três critérios: o tempo necessário para processar o respetivo algoritmo, o tamanho da sua chave e o nível de segurança necessário. Estes critérios estão interligados, uma vez que, como já foi referido anteriormente, o algoritmo e o tamanho de uma chave tem influência direta no nível de segurança da cifra/assinatura. A escolha destes fatores foi, então, estudada de modo a que os tempos de atraso fossem o mais reduzidos possível, sempre sem comprometer a segurança.

3.10 Implementação do protótipo

Nesta subsecção, são abordadas as ferramentas utilizadas para implementação do sistema proposto, com principal destaque para o simulador utilizado para prova de conceito. Além disso, são ainda apresentadas algumas decisões de implementação que poderão ter impacto nas simulações realizadas, bem como as suas respetivas justificações.

3.10.1 Ferramentas utilizadas

Nesta subsecção, apresenta-se uma breve explicação do simulador e da biblioteca criptográfica utilizados.

NS-3

O NS-3 é um simulador de redes orientado principalmente para fins educacionais e de pesquisa. Este simulador é *open-source* e está projetado de forma modular, o que permite que diferentes utilizadores desenvolvam novos protocolos e os partilhem pelo resto da comunidade, contribuindo para um maior leque de protocolos disponíveis para simulação [20].

Este simulador tem como base um conjunto de bibliotecas, cuja sua utilização pode ser combinada com bibliotecas externas. Isto permite a simulação do sistema proposto, uma vez que este faz uso de bibliotecas de criptografia, entre outras, que não fazem parte do simulador [20].

Linux é o principal sistema operativo para o qual este simulador está desenhado. Tem como linguagens base *Python* e C++, sendo necessário ter alguns conhecimentos numa destas para realizar as simulações pretendidas [20]. Para que seja possível fazer alterações ao nível dos protocolos implementados, efetivamente, é necessário saber a linguagem C++.

Para simular o sistema num ambiente controlado onde seja possível controlar as diferentes variáveis, quer para deteção de falhas ou obtenção de resultados como tempos de atraso, é necessário fazer uso de um simulador de redes. É pertinente que, neste simulador, seja possível utilizar protocolos já existentes, bem como implementar novos protocolos como é o caso do sistema proposto. Daí a escolha do software NS-3 para simulação deste sistema, pois este permite realizar as tarefas referidas anteriormente.

Libgcrypt

Para realizar as funções criptográficas inerentes a este sistema, decidiu recorrer-se à biblioteca *Libgcrypt*. Esta biblioteca funciona na maioria dos sistemas POSIX, é gratuita e contém funções que executam diversos algoritmos criptográficos, entre os quais encontram-se os que são utilizados neste sistema (referencia do manual da biblioteca). Permite, assim, a utilização de vários algoritmos dentro dos diferentes tipos (cifras simétricas, assimétricas e HMACs), para poder testar qual deles o mais indicado, fazendo face às características que se pretende que o sistema tenha [21].

3.10.2 Modelo implementado em NS-3

A presente subsecção tem como objetivo esclarecer o modo como o sistema foi implementado em NS-3. Isto revela-se importante, uma vez que é necessário entender algumas das limitações e constrangimentos que existiram, que potencialmente impediram uma melhor aproximação das simulações às condições reais. Do mesmo modo, são apresentadas algumas decisões de implementação que ajudam a essa mesma aproximação.

No contexto das limitações, a que possivelmente terá maior impacto no desempenho do sistema é o facto deste sistema estar implementado sobre o protocolo 802.11b ao menor ritmo binário disponível, 1Mbps. Tendo em consideração que o sistema foi projetado com o objetivo de operar principalmente em ambientes de natureza militar, onde o ritmo binário disponível é menor, os resultados das simulações podem revelar-se otimistas face à realidade. Importa ainda referir que todas as mensagens a circular na rede são enviadas em *broadcast*, não existindo assim garantia de entrega das mesmas por parte do protocolo 802.11b. Essa característica fica a cargo do sistema proposto neste capítulo.

Uma das decisões de implementação importantes é a introdução de um atraso aleatório no envio das mensagens, entre 0 e 2 segundos. Este atraso tem como objetivo evitar a colisão entre duas ou mais mensagens que seriam enviadas ao mesmo tempo, caso contrário. Por exemplo, dois ou mais nós

recebem uma mensagem de KA. Caso estes realizem um pedido de adesão ao grupo (*Join Request*), esses pedidos irão colidir. O atraso introduzido no envio das mensagens pretende evitar situações idênticas a esta, reduzindo o número de reenvios necessários e, por consequência, favorecendo um melhor desempenho do sistema. O intervalo de tempo para este atraso foi escolhido de modo a promover um menor reenvio de mensagens, caso contrário, para um elevado número de nós, o atraso na estabilização da rede seria ainda mais acentuado do que os 2 segundos previstos. Importa referir que, numa situação real, a probabilidade de estas colisões ocorrerem seria muito reduzida. Contudo, neste simulador, os eventos são todos sincronizados e os tempos de processamento de cada nó não são tidos em consideração, levando à ocorrência destas colisões.

De modo a que fosse possível uma melhor aproximação à realidade, os tipos de mensagens previstos originalmente no protocolo CRAN foram também implementados no sistema. Uma vez que estas mensagens irão circular pelo grupo, o impacto destas na rede poderá promover resultados mais fidedignos.

Uma das decisões que terá influência direta, não só na quantidade de bytes enviados, mas também no desempenho do sistema, é o tamanho de cada mensagem. Posto isto, na tabela 3.2, são apresentados os tipos de mensagens implementados, bem como os tamanhos dos respetivos campos em bytes.

Tabela 3.2: Tamanho dos diferentes tipos de mensagens em bytes.

	Tipo Msg.	ID Dest.	*Nonce*	Resp.	TS.	C. sim.	C. assim	Ass.	Cert.	Total
KA	1	-	-	-	4	16	-	66	203	290
Join Request	1	1	4	-	-	-	-	66	203	275
Join Reply	1	1	-	-	-	-	130	66	-	198
Leave Request	1	1	4	-	-	-	-	66	203	275
Leave Reply	1	1	4	1	-	-	-	66	-	73
Key Refresh	1	1	4	-	-	-	130	66	-	202
Link Request	1	-	-	-	-	16x	-	-	-	1+16x
Link Response	1	-	-	-	-	16x	-	-	-	1+16x
Update	1	-	-	-	-	16x	-	-	-	1+16x

Capítulo 4

Resultados de simulação

Neste capítulo serão apresentadas as simulações realizadas afim de testar o funcionamento do sistema proposto. Serão ainda expostos os resultados relativos a estas simulações, com principal foco nos tempos necessários para que o sistema fique consistente com a realidade, nas diferentes situações simuladas. Outra das métricas apresentadas, no que diz respeito aos resultados, é o número de mensagens e de bytes enviados pela rede, face ao número de nós pertencentes à mesma.

Para cada um dos diferentes cenários, foram realizadas 10 simulações com diferentes números de nós além do líder: 1, 5, 10, 20 e 30 nós. Doravante, importa reter que, sempre que a quantidade de nós for referida, irá ser sempre relativamente aos nós subalternos, sendo necessário contemplar ainda a existência do nó líder. Posto isto, para cada uma das combinações (cenário, nº de nós), foram realizadas 10 simulações de modo a que fosse possível retirar tempos por forma a obter o valor médio e o respetivo intervalo de confiança de 95%.

Ao longo de todas as simulações que serão apresentadas neste capítulo, definiram-se algumas condições que importam para uma melhor perceção dos resultados, a saber:

- A periodicidade do envio das mensagens de KA foi definida como 5 segundos;
- Os instantes em que cada nó envia o seu KA foram distribuídos uniformemente ao longo desse mesmo período (5 segundos), sendo que a ordem é definida por ordem crescente do ID de cada nó;
- O nó líder foi definido com o ID mais baixo, sendo por isso, o primeiro nó a enviar o seu KA em cada ciclo.

4.1 Formação inicial do grupo

Neste cenário, pretende-se avaliar os resultados obtidos para a situação inicial em que o grupo é formado, estando todos os nós participantes ao alcance uns dos outros. Os resultados retirados destas simulações dizem respeito ao período de tempo necessário para que seja eleito um líder e todos os outros nós tenham sido aceites no grupo do mesmo. As mesmas simulações foram realizadas

ainda para o modo *PFS*, sendo apresentados os resultados de ambas. De acordo com o número de nós existentes, a Figura 4.1 ilustra a disposição espacial dos mesmos, onde a vermelho se encontra representado o nó eleito para líder. Neste cenário específico, no qual todos os nós se encontram ao alcance uns dos outros, a sua disposição espacial não tem grande influência no resultado final.

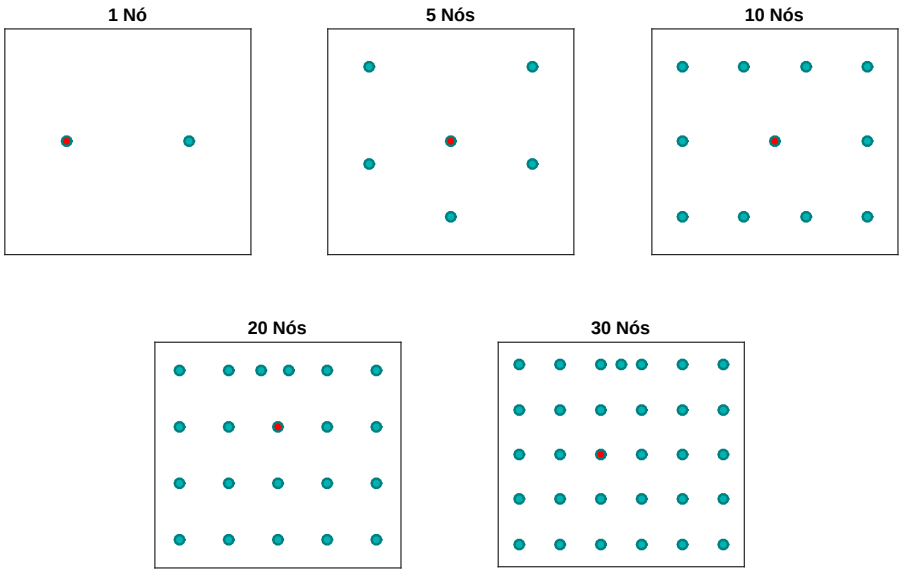

Figura 4.1: Disposição espacial dos nós (cenário 1 e 3).

Nas tabelas 4.1 e 4.2, encontram-se expostos os tempos obtidos de cada simulação, sem e com o modo *PFS* ativo, respetivamente. Como é possível observar, devido ao atraso aleatório introduzido no envio dos pedidos de adesão, os resultados das simulações encontram-se maioritariamente entre 0 e 2 segundos, mesmo em relação ao diferente número de nós. Ainda assim, à medida que o número de nós aumenta, ocorre com mais frequência a perda de mensagens, forçando o reenvio dos pedidos de adesão à rede. Isto justifica a existência de resultados que excedem o tempo de 2 segundos de atraso.

Tabela 4.1: Tempo, em segundos, registado para *PFS* inativo (cenário 1).

	Nº de nós				
Nº da simulação	1	5	10	20	30
1	2,0	2,0	1,8	2,7	2,7
2	0,8	2,0	1,8	1,9	3,0
3	1,1	1,6	1,7	2,0	2,6
4	1,3	1,8	1,8	2,4	2,7
5	1,3	1,5	1,7	1,8	3,8
6	1,1	2,0	2,2	2,0	2,5
7	0,3	1,9	1,6	2,7	3,4
8	1,4	1,9	2,0	1,9	2,3
9	1,7	1,6	1,7	2,6	3,5
10	2,0	1,7	1,9	2,1	2,9

Tabela 4.2: Tempo, em segundos, registado para *PFS* ativo (cenário 1).

Nº da simulação	Nº de nós				
	1	5	10	20	30
1	2,0	3,3	3,8	7,0	8,3
2	0,8	3,2	3,3	6,4	9,2
3	1,1	2,5	3,5	7,5	8,5
4	1,3	3,4	4,7	7,2	9,5
5	1,3	2,7	3,7	7,2	10,6
6	1,1	3,9	4,6	7,7	9,1
7	0,3	4,9	3,3	7,2	10,9
8	1,4	3,5	3,9	6,8	9,9
9	1,7	2,8	3,4	7,2	8,8
10	2,0	2,5	3,4	7,0	10,5

Como é possível observar pelos tempos obtidos, o funcionamento da rede em modo *PFS* é mais penoso para o sistema proposto. Esta diferença ganha mais notoriedade à medida que o número de nós aumenta. O gráfico presente na Figura 4.2 poderá ilustrar melhor a afirmação anterior, apresentando a média dos tempos obtidos nas simulações realizadas e os respetivos intervalos de confiança de 95%.

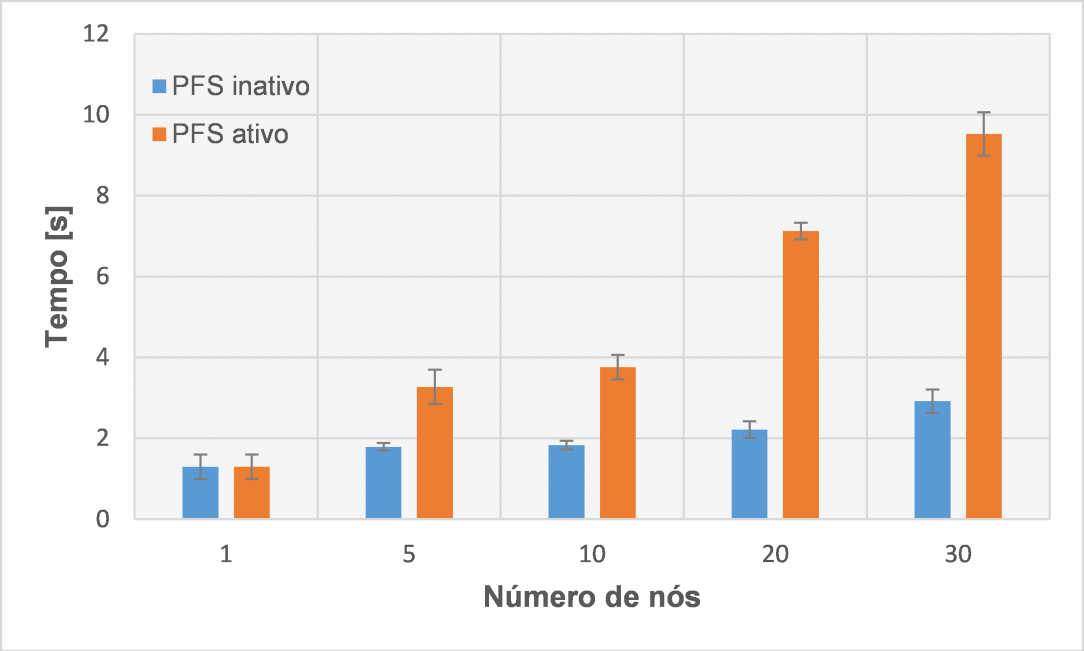

Figura 4.2: Gráfico da média de tempos e intervalos de confiança (cenário 1).

É possível ainda verificar através da tabela 4.3, que as situações onde o tempo de atraso excede os 2 segundos coincidem com as simulações nas quais foram perdidas mensagens, forçando o seu reenvio. A estas situações excluem-se os tempos de atraso que excedem por pouco os 2 segundos, uma vez que, para estes casos particulares, a mensagem já teria sido enviada pouco antes de acabarem os 2 segundos de atraso, tendo sido recebida apenas após estes terem decorrido. É ainda observável que, para os mesmos tamanhos de grupo, a quantidade de reenvios de pedidos de adesão é de uma forma

geral maior para o modo *PFS*. Isto deve-se à maior quantidade de mensagens a serem trocadas na rede.

Tabela 4.3: Número de reenvios de pedidos de adesão.

	PFS inativo					PFS ativo				
	Nº de nós					Nº de Nós				
Nº da simulação	1	5	10	20	30	1	5	10	20	30
1	0	0	0	2	3	0	4	6	89	193
2	0	0	0	0	2	0	5	8	55	126
3	0	0	0	0	2	0	2	12	61	219
4	0	0	0	1	2	0	4	11	59	198
5	0	0	0	0	8	0	2	19	105	133
6	0	0	1	0	1	0	3	13	87	110
7	0	0	0	1	4	0	2	10	85	188
8	0	0	0	2	2	0	4	16	46	235
9	0	0	0	1	7	0	2	10	64	122
10	0	0	0	0	5	0	1	17	54	166

4.2 Formação inicial do grupo (3 *hops*)

Os resultados a avaliar neste cenário são os mesmos relativamente ao cenário anterior. Contudo, a disposição espacial dos nós difere, isto porque o objetivo é que alguns destes se encontrem até 3 *hops* de distância do líder.

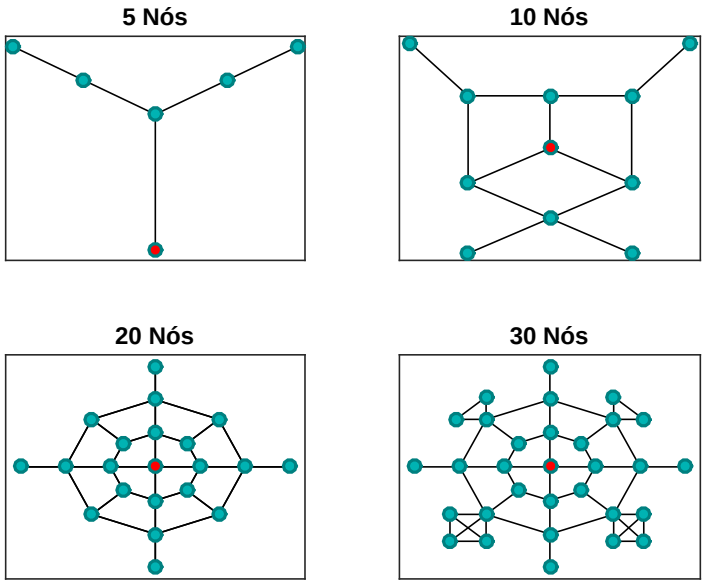

Figura 4.3: Disposição espacial dos nós (cenário 2).

A Figura 4.3 ilustra a disposição espacial dos nós, bem como os que se encontram ao alcance dos outros através das linhas representadas a preto. Não foram executadas simulações para 1 nó, uma vez que não seria possível obter-se 3 *hops*.

Tabela 4.4: Tempo, em segundos, registado para *PFS* inativo (cenário 2).

Nº da simulação	Nº de nós			
	5	10	20	30
1	13,7	14,0	9,2	9,7
2	9,9	13,2	9,7	11,8
3	8,0	8,7	4,1	8,7
4	8,0	12,9	9,9	9,3
5	13,9	8,1	9,3	6,8
6	9,4	14,3	8,7	8,6
7	12,6	8,6	9,3	8,2
8	10,0	13,5	9,6	7,9
9	8,7	9,0	8,4	9,0
10	5,8	14,0	9,0	8,5

Tabela 4.5: Tempo, em segundos, registado para *PFS* ativo (cenário 2).

Nº da simulação	Nº de nós			
	5	10	20	30
1	16,2	19,3	17,7	34,6
2	13,1	15,5	15,5	31,0
3	9,2	23,5	12,1	19,0
4	17,1	10,0	19,1	33,7
5	18,7	24,6	19,3	24,2
6	17,0	9,7	19,1	20,7
7	14,9	16,8	19,1	28,4
8	13,3	10,8	9,7	18,2
9	10,7	8,8	20,3	23,7
10	10,2	10,2	21,0	23,0

Nas tabelas 4.4 e 4.5, são apresentados os tempos obtidos em cada simulação para que os grupos sejam formados. Como é possível observar, os tempos obtidos são significativamente maiores em comparação com o cenário anterior (subsecção 4.1). Isto deve-se principalmente a duas razões. Uma delas é o aumento significativo do tráfego de mensagens na rede, uma vez que é exigido aos nós intermédios um esforço adicional no reencaminhamento dos pedidos de adesão. De outra forma, os nós que se encontram a 2 ou 3 *hops* nunca conseguiriam fazer chegar os seus pedidos ao líder do grupo. Com este aumento de tráfego, dá-se o aumento de mensagens perdidas por colisão, levando à necessidade de reenviar alguns pedidos de adesão que foram perdidos, aumentando assim o tempo para que o grupo se forme. A outra razão parte do simples facto de que um nó subalterno que se

encontre a 3 *hops* do líder só recebe a informação do mesmo após existir alguma rota na qual os nós que a formam já façam parte do grupo. Ou seja, em vez de serem feitos todos os pedidos de adesão ao mesmo tempo, como é o caso do cenário anterior, estes são feitos por camadas, começando pelos nós que se encontram diretamente ao alcance do líder e, em seguida, os nós que se encontram a 2 *hops* e assim sucessivamente.

Como seria expectável, os tempos obtidos são maiores quando o modo *PFS* está ativo, pelas razões já mencionadas no cenário anterior. Observando o gráfico da Figura 4.4, pode perceber-se que o impacto do modo *PFS* evidencia-se à medida que o número de nós aumenta.

Note-se ainda que, para as simulações com 5 e 10 nós, com *PFS* inativo, os tempos obtidos foram mais elevados do que para 20 e 30 nós. A razão para isto acontecer reside no facto de que a disposição espacial difere bastante e, consequentemente, tem influência no comportamento do sistema. No caso de 20 e 30 nós, como a disposição destes é bastante idêntica, os resultados obtidos também o são.

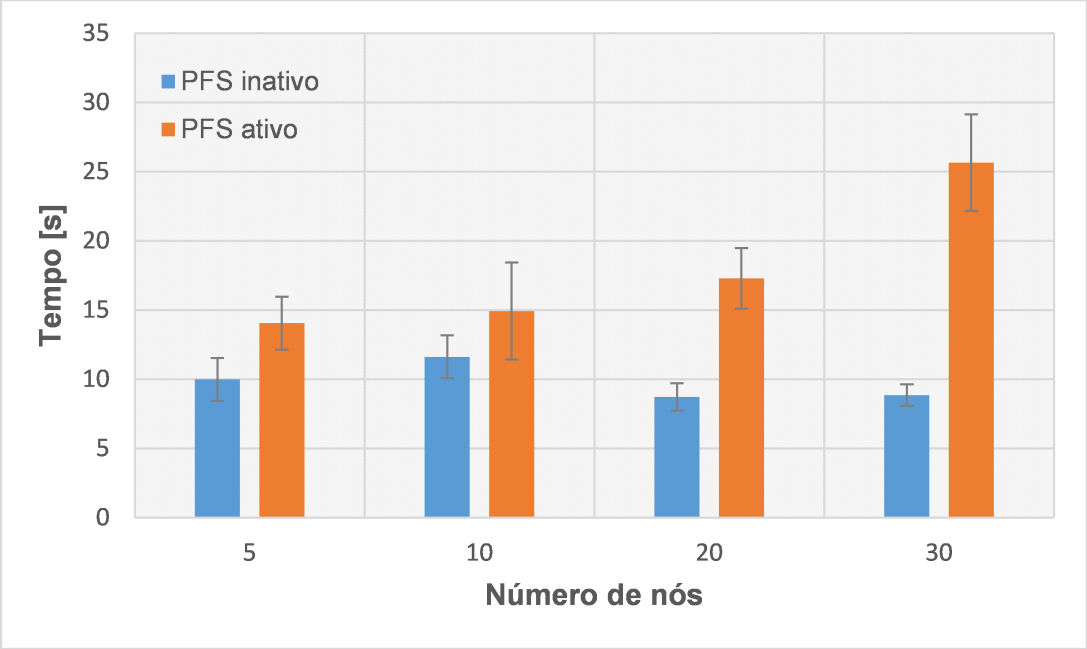

Figura 4.4: Gráfico da média de tempos e intervalos de confiança (cenário 2).

4.3 Saída de um nó do grupo

Neste cenário, pretende-se avaliar uma situação na qual, após o grupo estar totalmente formado, um dos nós sai do mesmo. O resultado principal que se pretende retirar destas simulações é o tempo necessário para que se verifiquem todas as seguintes condições:

- o nó saia do grupo;
- a chave de grupo seja renovada;
- a nova chave de grupo seja distribuída por todos os nós permanecentes.

Os nós encontram-se todos ao alcance uns dos outros. Para tal, a disposição dos nós foi a mesma em relação ao primeiro cenário (subsecção 4.1), podendo ser observada na Figura 4.1. O modo *PFS* não foi tido em consideração neste cenário, pois este apenas tem influência na adesão de nos à rede, não na saída destes.

Das tarefas necessárias para que o grupo volte a estabilizar após a saída de um dos nós, a mais exigente, em termos temporais, é a distribuição da nova chave de grupo pelos nós permanecentes. Como tal, à medida que o número de nós aumenta, também aumenta a quantidade de mensagens do tipo *KR* a serem enviadas pela rede. Isto leva a que, além de ser necessário mais tempo para que todas estas mensagens sejam enviadas, maior seja a probabilidade de colisão das mesmas, sendo assim exigido aos nós que não receberam a nova chave de grupo realizar pedidos de adesão. Como é percetível pelos resultados apresentados na tabela 4.6, quando esta situação ocorre cada vez com mais frequência, o tempo necessário para que a nova chave de grupo seja distribuída por todos os nós permanecentes aumenta. No caso das simulações realizadas para 30 nós, é notório o impacto que estas ocorrências têm na estabilização da rede.

Tabela 4.6: Tempo registado, em segundos (cenário 3).

Nº da simulação	Nº de nós			
	5	10	20	30
1	1,4	1,9	1,9	16,0
2	1,3	1,8	1,9	12,3
3	1,3	1,8	1,9	12,3
4	1,3	1,8	2,4	12,3
5	1,5	1,9	2,2	12,7
6	1,8	2,0	2,4	12,8
7	1,9	1,6	4,8	12,3
8	1,7	1,5	8,9	14,2
9	2,0	1,8	1,9	14,1
10	1,7	1,8	2,0	14,4

No que diz respeito aos resultados das simulações onde existe apenas 1 nó subalterno, os tempos obtidos são significativamente mais reduzidos. Isto deve-se ao facto de não ser necessário realizar a distribuição da nova chave de grupo, uma vez que o nó líder fica sem nós subalternos no grupo. Assim sendo, os resultados obtidos correspondem apenas ao tempo decorrido entre envio do pedido de saída por parte do nó subalterno e respetiva resposta. Por este motivo, no gráfico 4.5 não são apresentados os resultados da média e de intervalo de confiança para as simulações com 1 nó.

Neste cenário, os pedidos de saída realizados ao líder de grupo e respetivas respostas nunca foram perdidos, daí que não existiu nenhum reenvio desses pedidos.

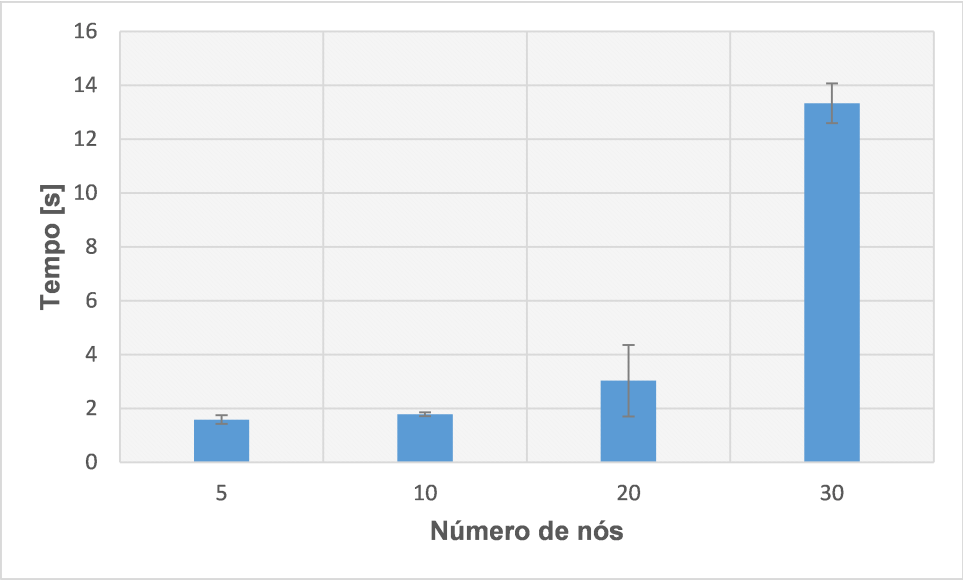

Figura 4.5: Gráfico da média de tempos e intervalos de confiança (cenário 3).

4.4 Partição do grupo

Esta subsecção aborda um cenário em que ocorrem alguns aspetos relevantes abordados em 3.7, nomeadamente a deteção de partição e a eleição de líder de grupo (3.7.3 e 3.7.4, respetivamente). Numa primeira fase, o cenário desenrola-se a partir de um grupo em que os nós que lhe pertencem encontram-se todos ao alcance uns dos outros, dispostos de forma igual ao primeiro cenário (subsecção 4.1). Posteriormente, dá-se um afastamento de um subconjunto desses nós, deixando-os fora de alcance.

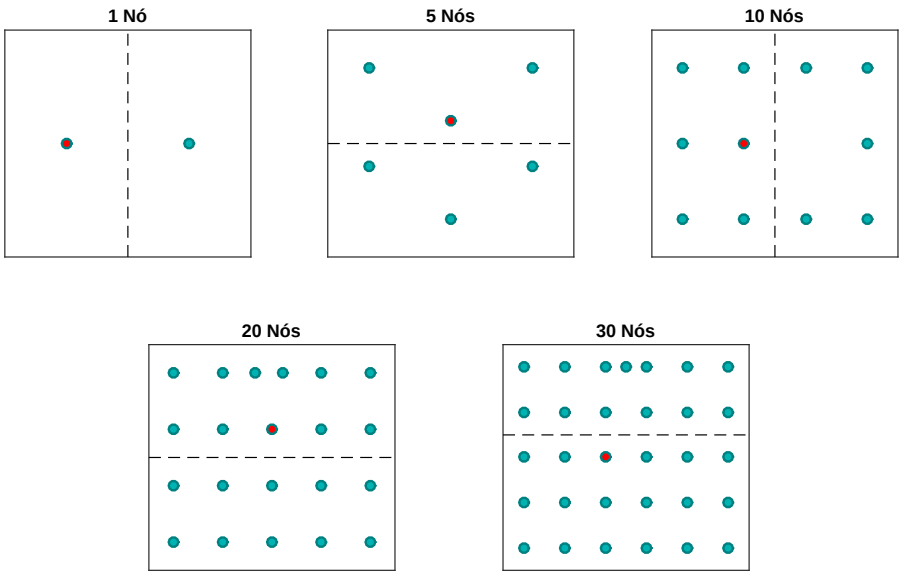

Figura 4.6: Disposição espacial dos nós (cenário 4 e 5).

Os principais resultados que se pretendem obter com esta simulação são os tempos necessários para que o subconjunto de nós, no qual não se encontra o líder, eleja um novo líder e forme o subgrupo. O cenário apresentado na subsecção 4.5 irá cobrir a junção destes dois subgrupos de volta ao grupo inicial. Na Figura 4.6, encontra-se a disposição espacial dos nós, a linha a tracejado representa a divisão dos dois subgrupos, após o afastamento dos nós.

Uma vez que a eleição do líder do novo subgrupo formado é interna a cada nó desse mesmo subgrupo, então o tempo necessário para que essa eleição ocorra é equivalente ao tempo necessário para que todos os nós subalternos desse subgrupo recebam o KA do novo líder.

Pode ocorrer a situação em que o KA do novo líder a ser eleito não seja recebido por todos os nós. Assim sendo, o tempo necessário para que o novo líder seja reconhecido como tal por todos os outros nós do subgrupo depende apenas do número de KAs a serem enviados e da sua periodicidade. De acordo com as simulações realizadas, não ocorreu nenhuma situação deste tipo, ou seja, todos os nós subalternos do subgrupo receberam o primeiro KA enviado pelo novo líder.

Conclui-se assim que o tempo necessário para que o novo subgrupo seja formado é apenas o tempo que decorre entre a ocorrência da partição e a deteção desta por parte do novo líder, mais o tempo de envio de um KA ($\sim$ 2,8ms). Contudo, o tempo que decorre até que a partição seja detetada não deve ser considerado, uma vez que, dependendo do instante real da partição do pelotão, o intervalo de tempo em que esta é detetada é sempre igual ou menor que período de envio dos KAs. Este tempo de atraso deve-se ao facto de o *timestamp* do líder original do grupo ainda não ter expirado.

Os resultados das simulações realizadas corroboram o que foi explicado acima. Estes resultados não são apresentados, uma vez que são idênticos para todas as simulações ao longo dos diferentes números de nós. Em suma, o tempo de eleição do novo líder poderá ser dado pela equação 4.1.

$$t_{eleição} = N_{reenvios\ KA} * T_{KA} + t_{envio\ KA} \tag{4.1}$$

Onde:

- $t_{eleição}$ é o tempo de eleição do novo líder;
- $N_{reenvios\ KA}$ é o número de envios de mensagens do tipo KA, além da primeira;
- T_{KA} é o período de envio de mensagens do tipo KA;
- $t_{envio\ KA}$ é o tempo de envio de um KA.

4.5 Junção do grupo

Partindo do estado final do cenário apresentado anteriormente (subsecção 4.4), no qual dois subgrupos se encontram fora de alcance, são apresentados os resultados obtidos em simulações nas quais esses mesmos grupos se voltam a juntar. Pretende obter-se os tempos necessários para que todo o grupo se volte a juntar, tendo todos o mesmo líder em comum, bem como a chave de grupo.

A disposição espacial dos nós pode ser observada na Figura 4.6, presente na subsecção anterior. Desta feita, a disposição final dos nós passa a ser a inicial e *vice-versa*.

Neste cenário prevêm-se duas situações distintas: uma em que a chave de grupo ainda é a mesma em ambos os subgrupos e outra na qual a chave difere. Na primeira situação referenciada, ocorre o mesmo que o explicado no cenário anterior (subsecção 4.4), ou seja, o tempo que decorre desde que todos os nós voltam a estar ao alcance uns dos outros e o grupo original seja reconstituído é equivalente ao tempo que decorre até que todos os nós subalternos recebam o KA do líder original. Esse tempo poderá, uma vez mais, ser representado pela equação 4.1. Os resultados das simulações, mais uma vez, estão em concordância com a explicação anterior.

No que diz respeito aos resultados obtidos na segunda situação, é de esperar que estes sejam de certa forma idênticos aos apresentados na subsecção 4.1, isto porque o processo de junção dos dois subgrupos é semelhante ao de formação inicial do grupo, uma vez que ambos se resumem à executam de pedidos de adesão ao grupo. Contudo, a diferença encontra-se no número de nós. Na subsecção 4.1, todos os nós fazem pedidos de adesão ao grupo. Já neste caso, apenas os nós do subgrupo que ficou sem o líder original o fazem. Assim sendo, para os diferentes números de nós presentes neste cenário, 1, 5, 10, 20 e 30, apenas são executados pedidos de adesão por parte de 1, 3, 5, 10 e 13 deles, respetivamente. Na tabela 4.7, é possível observar que os resultados obtidos estão de acordo, não só com o que foi referido anteriormente, mas também com os resultados apresentados na subsecção 4.1 (tendo em consideração a quantidade de pedidos de adesão que são realizados).

Tabela 4.7: Tempo registado, em segundos (cenário 5).

		Nº de nós				
		1	5	10	20	30
Nº da simulação	1	1,0	1,1	1,7	0,9	2,0
	2	0,6	1,2	1,7	1,1	2,0
	3	1,8	0,5	2,0	1,0	2,0
	4	0,9	0,5	1,5	1,6	2,1
	5	0,4	0,1	1,7	1,0	2,5
	6	1,8	0,9	1,1	0,9	1,8
	7	2,0	1,1	2,0	1,3	3,7
	8	1,0	1,2	1,9	0,9	2,6
	9	1,7	1,0	1,9	0,9	2,1
	10	1,0	1,0	1,2	1,0	2,9

Apresenta-se, ainda, no gráfico da Figura 4.7, a média dos tempos obtidos para os diferentes tamanhos de grupo, bem como os respetivos intervalos de confiança de 95%. Como, de uma forma geral, todos os resultados se encontram dentro dos 2 segundos de atraso que é introduzido aos pedidos de adesão, é aceitável que os tempos das simulações não aumentem de acordo com o número de nós, uma vez que a componente aleatória ainda tem uma forte influência nos resultados.

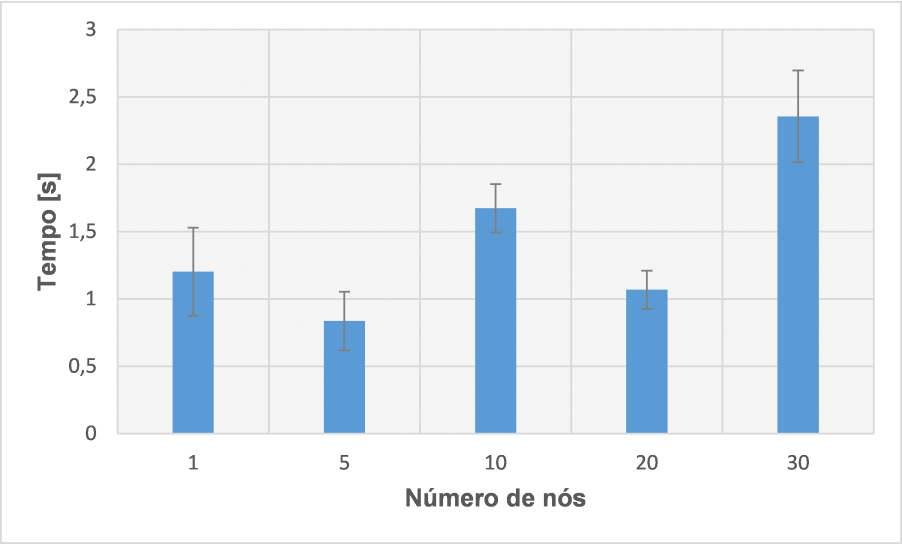

Figura 4.7: Gráfico da média de tempos e intervalos de confiança (cenário 5).

Capítulo 5

Conclusões

Neste capítulo apresentam-se as conclusões retiradas ao longo do desenvolvimento da presente dissertação. Encontra-se dividido em duas partes: considerações finais relativamente ao estudo desenvolvido e propostas de trabalhos futuros.

5.1 Considerações finais

A disponibilidade de informação nos TO atuais é de extrema importância no auxílio à tomada de decisão por parte dos comandantes. Daí a existência de sistemas de gestão do campo de batalha em tempo real, no qual o CRAN é um dos protocolos de suporte, permitindo fazer chegar essas informações ao comandante e/ou disseminá-las pelos elementos da força em questão. Por isso mesmo, estes sistemas são considerados alvos remuneradores por parte do inimigo, uma vez que o acesso por parte deste permite-lhe obter informações acerca das nossas forças. O sistema desenvolvido nesta dissertação visa, em primeira instância, negar ao inimigo a obtenção de informação acerca das forças que utilizam o protocolo CRAN. Visa ainda, negar a disrupção dos sistemas de gestão do campo de batalha, através de ataques ao protocolo CRAN, sejam estes por *Denial of Service* ou pela introdução de falsas informações na rede.

Pretendeu-se, com o sistema proposto nesta dissertação, que este conferisse segurança ao protocolo CRAN através da distribuição de chaves pela rede e do uso destas para cifrar as comunicações. A um nó que faça parte do grupo que, por consequência, tenha conhecimento da chave de grupo, é possível cifrar e decifrar as mensagens do protocolo CRAN. Para tal, foram definidas as características de adição, remoção e expulsão de nós do grupo, de modo a que exista controlo no acesso ao grupo. Este controlo é exercido por um líder, que é eleito dinamicamente, tendo como critério a sua patente (indicador da sua precedência relativamente à liderança). Foi ainda tida em consideração a necessidade de que duas ou mais unidades possam coexistirem em simultâneo, utilizando este sistema, sem que interferissem mutuamente nas suas comunicações. A capacidade do sistema se adaptar à destruição de qualquer nó e à partição dos grupos em subgrupos, foram outros dos desafios da presente dissertação, ambos cumpridos pela combinação dos conceitos de deteção de partição, eleição de líder

e *rejoin*. Ao longo do desenvolvimento deste sistema, procurou-se sempre garantir um compromisso entre a segurança da rede e o tráfego de mensagens na mesma. Isto porque o sistema deve operar, principalmente, em ambiente militar, onde a utilização de equipamentos, tais como o rádio PRC-525, faz com que largura de banda disponível seja habitualmente reduzida. Posto isto, um dos pontos críticos residiu na necessidade de enviar os certificados em algumas das mensagens e no facto de estes terem um tamanho avultado quando comparado com o restante conteúdo das mensagens. Foi por esta razão que se recorreu a conceitos alternativos de certificados, cujos tamanhos são reduzidos, quando comparados com os certificados padrão usualmente utilizados.

A revisão do estado da arte revelou-se importante na aquisição de conhecimentos e aplicação dos mesmos no desenvolvimento do sistema proposto, permitindo ter uma noção alargada dos conceitos existentes e comummente utilizados nesta área. Com isto, de entre todos estes conceitos, foi possível aplicar os mais adequados, tendo em consideração as limitações e por forma a atingir os objetivos delineados.

Embora o sistema proposto tenha sido desenhado tendo em vista as características do protocolo CRAN, este consegue ser independente do mesmo, uma vez que a única ligação entre o funcionamento de ambos encontra-se na utilização do tipo de mensagem KA como *beacon*. Esta ligação é facilmente quebrada separando os campos relativos ao *beacon* que lhe foram adicionados, enviando-os à parte. Isto permite que este sistema possa eventualmente ser utilizado para garantir segurança de outros protocolos com comportamentos similares ao nível de redes Ad-hoc.

As simulações realizadas tiveram em consideração diferentes cenários, nos quais foi possível testar os conceitos definidos para cumprimento dos objetivos propostos. Para cada um destes cenários foram realizadas simulações com diferentes números de nós, isto para que se pudesse apreciar o impacto do número de nós no desempenho do sistema. Destas simulações, verificou-se, principalmente, um aumento dos tempos obtidos e do tráfego de mensagens na rede à medida que o tamanho do grupo aumentava. Este aumento é ainda mais evidente nas simulações que foram realizadas com o modo *PFS* ativo, onde, além dos tempos obtidos, também cresceu a quantidade de mensagens perdidas e, consequentemente, o número de reenvios.

5.2 Trabalhos futuros

A possibilidade de realizar testes em ambiente real, implementando efetivamente o sistema nas unidades do exercito e analisar o impacto deste nas comunicações e coerência da imagem operacional (*Common Operational Picture*, COP), seria um importante desenvolvimento na sequência desta dissertação. Note-se que desta feita, a largura de banda disponível é mais reduzida relativamente à utilizada para as simulações nesta dissertação. Ainda neste contexto, o conceito de chave de sessão, apresentado anteriormente na subsecção 3.2.7, poderá ser desenvolvido, caso se verifique essa necessidade.

Um outro trabalho interessante para futuro seria conciliar as tabelas de vizinhos e de topologia do protocolo CRAN para que mais facilmente sejam reencaminhadas as mensagens definidas nesta

dissertação, evitando assim que todos os nós reencaminhem as mensagens. Assim, apenas os nós com métrica mais próxima do líder o fariam, reduzindo a quantidade de dados a ser transmitidos na rede. Em vez de se ter a segurança separada do próprio CRAN, ter-se-ia ambos a trabalhar em conjunto de forma potencialmente mais eficiente.

Tendo em vista aumentar a versatilidade do sistema proposto, seria interessante estudar a possibilidade dos certificados dos nós ou certificados de revogação serem recebidos via outros métodos durante o funcionamento da rede. Este estudo iria remover a necessidade de cada nó receber o seu certificado por parte da AC antes da inicialização da rede. Possibilitando assim, a adição de um determinado nó ao grupo durante o funcionamento da rede, mesmo que, tal não se previsse inicialmente e não fosse possível que esse nó recebesse o certificado diretamente da AC. Os certificados de revogação devem ser contemplados, uma vez que, a qualquer momento durante o funcionamento da rede, qualquer nó pode ser capturado pelo inimigo. Face a esta situação, deve ser impedida a participação deste nó na rede através da emissão destes certificados de revogação, mesmo que os líderes de grupo tenham capacidade para o expulsar dos seus grupos a qualquer momento.

Outra das áreas onde é proposta a realização de um trabalho futuro reside no estudo dos tempos necessários para cifrar, decifrar, assinar e comprovar a validade das assinaturas das mensagens. Estes tempos dependem essencialmente do tipo de cifra ou assinatura e do poder de processamento disponível no respetivo nó. Daí que este trabalho proposto deveria incidir sobretudo no impacto destes tempos no funcionamento do sistema apresentado e ainda na recomendação de determinado poder de processamento gerindo o compromisso entre os custos e o impacto referido.

Bibliografia

[1] G. Gomes, A. Zúquete, and S. Sargento. Self-adapted protocol for intra and inter-echelons communications. In *International Telecommunications Network Strategy and Planning Symposium - networks*, volume 1, pages 1 – 1, June 2014.

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

[3] A. Amodei Júnior and O. C. Duarte. Segurança no roteamento em redes móveis ad hoc. In *Seminário de Tópicos Especiais em Redes de Computadores*, 2003. GTA - Universidade Federal do Rio de Janeiro.

[4] N. C. Fernandes. Análise de ataques e mecanismos de segurança em redes ad hoc. Master's thesis, Universidade Federal do Rio de Janeiro, Escola Politécnica, Dezembro 2006.

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

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

[7] M. Nyström and J. Brainard. An x. 509-compatible syntax for compact certificates. In *Secure Networking—CQRE [Secure]'99*, pages 76–93. Springer, 1999.

[8] A. L. d. S. Eduardo da Silva. Implementação de um esquema de gerenciamento de chaves auto-organizado para redes ad hoc móveis, 2007.

[9] L. Zhou, L. Z. Department, and Z. J. Haas. Securing ad hoc networks, 1999. Cornell University.

[10] S. Capkun, L. Buttyán, and J.-P. Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE TRANSACTIONS ON MOBILE COMPUTING*, 2(1):52–64, 2003.

[11] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theor.*, 22(6):644–654, Sept. 2006. ISSN 0018-9448. doi: 10.1109/TIT.1976.1055638. URL `http://dx.doi.org/10.1109/TIT.1976.1055638`.

[12] M. Steiner, G. Tsudik, and M. Waidner. Diffie-hellman key distribution extended to group communication. In *ACM Conference on Computer and Communications Security*, pages 31–37, 1996. URL `citeseer.ist.psu.edu/steiner96diffiehellman.html`.

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

[14] D. Dolev and A. C. Yao. On the security of public key protocols. Technical report, Stanford, CA, USA, 1981.

[15] J. Herzog. A computational interpretation of dolev-yao adversaries. *Theor. Comput. Sci.*, 340(1): 57–81, June 2005. ISSN 0304-3975. doi: 10.1016/j.tcs.2005.03.003. URL http://dx.doi.org/10.1016/j.tcs.2005.03.003.

[16] F. Swiderski and W. Snyder. *Threat modeling*. Redmond, Wash. Microsoft Press, 2004. ISBN 0-7356-1991-3. URL http://opac.inria.fr/record=b1119864.

[17] S. Hernan, S. Lambert, T. Ostwald, and A. Shostack. Uncover security design flaws using the stride approach, Novembro 2006. URL msdn.microsoft.com.

[18] M. O. Pervaiz, M. Cardei, and J. Wu. *Network Security*, chapter Routing Security in Ad Hoc Wireless Networks, pages 117–142. Springer US, Boston, MA, 2010. ISBN 978-0-387-73821-5. doi: 10.1007/978-0-387-73821-5_6. URL http://dx.doi.org/10.1007/978-0-387-73821-5_6.

[19] C. F. Kerry, A. Secretary, and C. R. Director. Fips pub 186-4 federal information processing standards publication digital signature standard (dss), 2013.

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

[21] W. Koch and M. Schulte. The libgcrypt reference manual. https://gnupg.org/documentation/manuals/gcrypt/gcrypt.pdf, Setembro 2015. Acedido: 12-08-2016.