EDUARDO DA SILVA

# SEMAN - UMA PROPOSTA DE MIDDLEWARE SEGURO PARA AS REDES AD HOC MÓVEIS

Tese apresentada como requisito parcial à obtenção do grau de Doutor. Programa de Pós-Graduação em Ciência da Computação, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Prof. Dr. Luiz Carlos Pessoa Albini

CURITIBA

2014

EDUARDO DA SILVA

# SEMAN - UMA PROPOSTA DE MIDDLEWARE SEGURO PARA AS REDES AD HOC MÓVEIS

Tese apresentada como requisito parcial à obtenção do grau de Doutor. Programa de Pós-Graduação em Ciência da Computação, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Prof. Dr. Luiz Carlos Pessoa Albini

CURITIBA

2014

S586s Silva, Eduardo da
SEMAN - uma proposta de middleware seguro para as redes ad hoc móveis / Eduardo da Silva. – Curitiba, 2014.
205f. : il. [algumas color.], tab.

Tese (doutorado) - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-graduação em Ciência da Computação, 2014.

Orientador: Luiz Carlos Pessoa Albini
Bibliografia: p. 169-185.

1. Sistemas de comunicação móvel. 2. Middleware. I. Albini, Luiz Carlos Pessoa. II. Universidade Federal do Paraná. III. Título.

CDD: 004.6

Ministério da Educação
Universidade Federal do Paraná
Programa de Pós-Graduação em Informática

# PARECER

Nós, abaixo assinados, membros da Banca Examinadora da defesa do aluno de Doutorado em Ciência da Computação, Eduardo da Silva, avaliamos a tese de doutorado intitulada *"Seman - uma proposta de middleware seguro para redes ad hoc móveis"*, cuja defesa pública foi realizada no dia 04 de abril de 2014, às 14:00 horas, no Departamento de Informática do Setor de Ciências Exatas da Universidade Federal do Paraná. Após avaliação, decidimos pela:

(X) **aprovação** do candidato. ( ) **reprovação** do candidato.

Curitiba, 04 de abril de 2014.

Prof. Dr. Luiz Carlos Pessoa Albini
**DINF/UFPR – Orientador**

Profa. Dra. Regina Borges de Araujo*
**UFSCAR – Membro Externo**

Prof. Dr. Routo Terada
**USP – Membro Externo**

Prof. Dr. Carlos Alberto Maziero
**UTFPR – Membro Interno**

Prof. Dr. Eduardo Cunha de Almeida*
**DINF/UFPR – Membro Interno**

* Participação por videoconferência

# AGRADECIMENTOS

Quero agradecer, antes de tudo, a Deus, Pai e Filho e Espírito Santo, pelo dom da vida e pela graça proporcionada de realizar este árduo trabalho. Obrigado, ó Deus imenso, por dar-me forças nos momentos de fraqueza, iluminar-me nas horas de incertezas e suprir todas as minhas necessidades.

Estendo os meus agradecimentos à minha pérola preciosíssima: minha família. A conclusão deste trabalho não teria o mesmo sabor se ele não tivesse sido acompanhado pelo carinho e apoio dos que eu mais amo. Agradeço à minha amada esposa Jesli, que é a minha fortaleza, ânimo e alegria, em todos os momentos de nossa vida. Obrigado, meu amor, por dedicar-se, tão belamente, pela união e construção do nosso lar. Agradeço também aos meus amados filhos, Miguel e Lorena. Vocês proporcionam as melhores horas dos nossos dias e os momentos mais alegres da nossa casa.

Quero agradecer também a todos os meus familiares, de modo especial à minha mãe, Maria Terezinha, pelo seu exemplo e por toda a sua dedicação na criação dos seus filhos. Também ao carinho e apoio dos meus irmãos Marcos, Thiago e Vanessa. Obrigado a cada um de vocês, por serem presença constante nas nossas vidas.

Agradeço ao meu orientador, Albini, por ter confiado em mim e ter me ensinado o caminho da ciência ao longo desses anos. Você tem me ensinado muito durante todo esse processo de orientação, não apenas quanto à pesquisa, mas também quanto à postura de um grande professor diante de seus alunos. Além disso, foi um verdadeiro amigo nos vários momentos, bons e ruins, que vivi no mestrado e no doutorado. Agradecendo ao prof. Albini, agradeço também todos os professores do Departamento de Informática. Um agradecimento especial aos que estiveram envolvidos, direta ou indiretamente na minha formação: Aldri, Michele, André [Guedes], [André] Vignatti, Eduardo [Almeida], Luiz [Oliveira], e Roberto. Muito obrigado, a cada um de vocês, pela colaboração na minha formação. Agradeço também a todos os membros do grupo NR2: é muito bom fazer parte de um grupo como esse.

Por fim, agradeço aos membros da minha banca de defesa: profa. Regina (UFSCar), prof. Routo Terada (USP), prof. Carlos Maziero (UTFPR), prof. Eduardo Almeida (UFPR), prof. Roberto (UFPR). As contribuições de cada um de vocês foi essencial para deixar este trabalho melhor. Muito obrigado pela seriedade na leitura e sugestões de melhorias e pelo incentivo na busca das melhores práticas de atuação de um pesquisador.

Muito obrigado!

*Que Deus me conceda falar com inteligência e ter pensamentos dignos dos dons que recebi, pois é Ele quem guia a Sabedoria e dirige os sábios. Nós estamos nas suas mãos, nós e nossos discursos, toda a nossa inteligência e nossa habilidade.*

Sabedoria 7,15-16

# RESUMO

Devido às particularidades das redes ad hoc móveis (MANETs - *Mobile Ad Hoc Networks*), como a topologia dinâmica, a ausência de infraestrutura e a sua característica decentralizada, a implementação de aplicações complexas e flexíveis para estas redes torna-se um desafio. Para permitir o desenvolvimento dessas aplicações, diversas soluções de *middleware* foram propostas. Contudo, as soluções encontradas não consideram plenamente os requisitos de segurança dessas redes. Este trabalho apresenta um estudo dos *middlewares* propostos para as MANETs, relatando o seu funcionamento e apresentando um comparativo das funcionalidades disponíveis. Esses *middlewares* são categorizados de acordo com a seguinte classificação, proposta neste trabalho: baseados em espaços de tuplas, baseados em P2P, baseados em contexto, *cross-layer* e orientados à aplicação. Em seguida, com base nas limitações estudadas, é proposto um novo *middleware* de segurança para as MANETs, chamado de *SEcure Middleware for Ad hoc Mobile Networks* (SEMAN - *Middleware* seguro para as redes ad hoc móveis), que fornece um conjunto de serviços de segurança para facilitar o desenvolvimento de aplicações distribuídas, complexas e flexíveis. Para fornecer tais serviços e garantir a segurança, o SEMAN considera o contexto das aplicações e organiza os nós em grupos, também baseados nesses contextos. O *middleware* prevê três módulos: serviço, processamento e segurança. O módulo de serviço é responsável por manter todos os serviços e aplicações que são disponibilizados pelo nó hospedeiro a outros nós da rede. O módulo de processamento é responsável por manter o funcionamento central do *middleware*, atendendo os pedidos e gerenciando o registro dos serviços e componentes disponíveis. O módulo de segurança é o ponto principal do *middleware* e o foco desta tese. Ele possui os componentes de gerenciamento de chaves, de confiança e de grupos. Todos esses componentes foram desenvolvidos pelo autor e são descritos neste trabalho. Eles são suportados por um núcleo de operações criptográficas e atuam de acordo com regras e políticas de segurança. A integração desses componentes fornece garantias de segurança contra ataques às aplicações que utilizam o *middleware*.

# ABSTRACT

Due to the particularities of Mobile Ad Hoc Networks (MANETs), as their dynamic topology, lack of infrastructure and decentralized characteristic, the implementation of complex and flexible applications is a challenge. To enable the deployment of these applications, several middleware solutions were proposed. However, these solutions do not completely consider the security requirements of these networks. This thesis presents middleware solutions for MANETs, by describing their operations and presenting a comparative of the available functionalities. The middlewares were grouped according to this classification: tuple space-based, P2P-based, context-based, cross-layer and application-oriented. Then, based on the limitations of the studied solutions, a new secure middleware is proposed, called *SEcure Middleware for Ad hoc Networks* (SEMAN), which provides a set of basic and secure services to MANETs aiming to facilitate the development of distributed, complex and flexible applications. To provide such services and ensure security to the applications, SEMAN considers the context of applications and organizes nodes into groups, also based on these contexts. The middleware includes three modules: service, processing, and security. Service module is responsible for maintaining all services and applications hosted by nodes. The processing module is responsible for maintaining the middleware core operation, listening the requests and managing the registry of available services and components. The security module is the main part of the middleware and the focus of this thesis. It has the following components: key management, trust management and group management. All these components were developed and are described in this work. They are supported by a cryptographic core and behave according to security rules and policies. The integration of these components provides security assurance against attacks to the applications that use the middleware.

# SUMÁRIO

# LISTA DE ILUSTRAÇÕES

# LISTA DE TABELAS

# LISTA DE ABREVIATURAS E SIGLAS

***i*FUSO** *Identity-Based Fully Self-Organized Key Management for MANETs. 118, 120–127, 132–135, 153, 156, 158*

**ABM** *Anti-Blackhole Mechanism. 198*

**AES** *Advanced Encryption Standard. 191*

**AGAPE** *Allocation and Group Aware Pervasive Environment. 47, 48*

**AODV** *Ad hoc On-Demand Distance Vector. 193, 195, 197*

**API** *Application Programming Interface. 26, 31, 34, 38, 41, 50, 51*

**ARAN** *Authenticated Routing for Ad hoc Networks. 200*

**CA** *Certificate Authority. 114*

**CAST** *Coordination Across Space & Time. 33, 34*

**CLDC** *Connected Limited Device Configuration. 41*

**COCA** *COoperative CAching. 54, 55, 59*

**CRC** *Ciclic Redundancy Check. 190*

**CTS** *Clear To Send. 189*

**D-PKG** *Distributed PKG. 114–116, 118, 120–123, 125, 127, 129–131, 134, 135, 147, 155, 157, 159–161*

**DPRAODV** *Detection, Prevention and Reactive AODV. 197*

**DSR** *Dynamic Source Routing. 33, 193*

**DSSS** *Direct Sequence Spread Spectrum. 188*

**EAP** *Extensible Authentication Protocol. 192*

**EMMA** *Epidemic Messaging Middleware for Ad hoc networks. 46, 52*

**FAP** *Flooding Attack Prevention. 195*

**FHSS** *Frequency Hopping Spread Spectrum. 188*

**IBC** *Identity-Based Cryptosystem. 79–81, 83, 85, 86, 114, 204, 205*

**IBE** *Identity-Based Encryption. 85, 204*

**IBS** *Identity-Based Signature. 85, 204*

**IDS** *Intrusion Detection System. 198*

**IKM** *Identity-based Key Management. 117*

**IP** *Internet Protocol. 49, 80, 85, 204*

**ITS** *Interface Tuple Space. 30*

**J2ME** *Java 2 Mobile Environment. 40–42*

**JMS** *Java Message Service. 46, 62*

**JNDI** *Java Naming Discovery Interfaces. 62, 65*

**JVM** *Java Virtual Machine. 47*

**JXME** *JXTA for Micro Edition. 40*

**LIME** *Linda In a Mobile Environment. 29, 30, 36*

**LP$^3$** *Limiting Packet Propagation Parameter. 199*

**MANET** *Mobile Ad Hoc Network. 20–24, 27, 29, 31, 33, 35–43, 46, 47, 53, 57, 60–62, 64, 65, 67, 68, 70, 73, 76, 80, 81, 83, 84, 86, 88, 89, 95–100, 103, 114, 115, 121, 123, 125, 135, 137–140, 164, 165, 187, 189, 190, 192–195, 197–202, 205*

**MIDP** *Mobile Information Device Profile. 41*

**NIST** *National Institute of Standards and Technologies. 83*

**P2P** *Peer-to-Peer. 22, 36, 37, 41, 42, 55*

**PASMi** *Photo Annotation and Sharing Middleware. 62, 63*

**PKG** *Private Key Generator. 80, 84, 86, 89, 114–117, 121, 130, 131, 161, 165, 204, 205*

**PKI** *Publick Key Infrastructure. 85, 114, 204*

**QAM** *QoS-aware Adaptive Middleware. 51, 52*

**QoS** *Quality of Service. 29, 51*

**RADIUS** *Remote Authentication Dial In User Service. 192*

**REDMAN** *REplication in Dense MANETs. 59–61*

**RERR** *Route Error. 194*

**RREP** *Route Reply. 193*

**RREQ** *Route Request. 193*

**RTS** *Request To Send. 189*

**SAODV** *Secure AODV. 200*

**SELMA** *Self-organized Marketplace-based Middleware for MANETs. 44, 45*

**SEMAN** *SEcure Middleware for mobile Ad hoc Networks. 23–25, 67–70, 72, 75–79, 84, 87–92, 94, 97, 109, 113, 118, 135, 139, 140, 143, 147, 149, 151, 152, 154, 156, 159–165, 167, 201*

**SHA** *Secure Hash Algorithm. 83*

**STEAM** *Scalable Timed Events And Mobility. 43, 44*

**TKIP** *Temporal Key Integrate Protocol. 191*

**TOTA** *Tuples On The Air. 31, 36*

**TRUE** *TRUst Evaluation service for MANETs. 96, 100, 113, 153, 156, 158*

**TrustUM** *Trust with UltimatuM game. 97, 109, 110, 113*

**TTL** *Time-To-Live. 199*

**UDP** *User Datagram Protocol. 49*

**WEP** *Wired Equivalent Privacy. 190, 191*

**WPA** *Wi-fi Protected Access. 190, 191*

# LISTA DE SÍMBOLOS E NOTAÇÕES

| | |
|---|---|
| $N_i$ | identidade do nó $i$ |
| $SK_i$ | chave privada do nó $i$ |
| $PK_i$ | chave pública do nó $i$ |
| $\mathbb{G}_1$ | grupo aditivo cíclico de ordem prima $p$ |
| $\mathbb{G}_2$ | grupo multiplicativo cíclico de ordem prima $p$ |
| $\zeta$ | tamanho em *bit* de um texto plano |
| $e$ | um emparelhamento bilinear em que $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ |
| $H_1(x)$ | função *hash* em que $H_1(x) = \{0,1\}^* \rightarrow \mathbb{G}_1^*$ |
| $H_2(x)$ | função *hash* em que $H_2(x) = \mathbb{G}_2 \rightarrow \mathbb{Z}_p^*$ |
| $H_3(x)$ | função *hash* em que $H_3(x) = \mathbb{G}_2 \rightarrow \{0,1\}^\zeta$ |
| $N_\mathcal{F}$ | nós fundadores |
| $MSK$ | chave privada mestre do sistema |
| $MPK$ | chave pública mestre do sistema |
| $MSK_i$ | parte da chave privada mestre mantida pelo nó $i$ |
| $TV_{(N_x,N_v)}$ | valor de confiança do nó $N_x$ no nó $N_v$ |
| $TC^x_{(N_x,N_v)}$ | cadeia de confiança $x$ do nó $N_x$ para o nó $N_v$ |
| $G_{tr}$ | grafo da rede confiança baseada em contexto |
| $G^x_{tr}$ | grafo da rede confiança baseada em contexto do nó $N_x$ |
| $N_a \rightarrow N_b$ | nó $N_a$ confia no nó $N_b$ |
| $\Delta_T$ | intervalo entre as atividades de monitoramento de vizinhos |
| $\Delta T_{ex}$ | intervalo entre as trocas de informação |
| $\alpha$ | limite das trocas de informação |
| $\beta$ | limite das avaliações de confiança |
| $G_\alpha$ | identificação do grupo $\alpha$ |
| $GEK_\alpha$ | chave de cifração do grupo $\alpha$ |
| $GDK_\alpha$ | chave de decifração do grupo $\alpha$ |
| $Sign_i$ | parte da assinatura de grupo chave mantida pelo nó $i$ |
| $\|Z\|$ | tamanho de um dado conjunto $Z$ |
| $a\|\|b$ | informação $a$ concatenada com informação $b$ |
| $a \oplus b$ | ou exclusivo de $a$ e $b$ |
| $\cong$ | aproximadamente |

# CAPÍTULO 1

# INTRODUÇÃO

As Redes *Ad Hoc* Móveis (*Mobile Ad Hoc Network*s(MANETs)) são formadas por um conjunto de dispositivos móveis (nós) que se comunicam entre si usando um canal de comunicação sem fio. Essas redes são estabelecidas dinamicamente sem depender de uma infraestrutura fixa ou uma administração centralizada e o seu funcionamento é mantido pelos próprios nós de uma forma auto-organizada (PAPADIMITRATOS; HAAS, 2005). A topologia é dinâmica, pois os nós podem se movimentar livremente pelo ambiente e podem entrar e sair da rede a qualquer momento sem notificarem uns aos outros.

Estas características tornam as MANETs atrativas para diversos cenários, principalmente quando a implantação de infraestrutura de comunicação é difícil ou o custo é muito alto (CHLAMTAC; CONTI; LIU, 2003). Alguns exemplos de aplicações dessas redes são (WU et al., 2006): soldados transportando informações sobre o campo de batalha; pessoas compartilhando informações durante uma reunião; participantes usando *notebooks* em uma conferência interativa; equipes de resgaste trabalhando após desastres como incêndio, furacão ou terremoto.

Este capítulo apresenta uma contextualização do tema a ser desenvolvido neste trabalho. Depois, são descritos os objetivos, a metodologia utilizada, as contribuições e a organização da tese.

## 1.1 Contextualização

As MANETs possuem diversos desafios resultantes das suas características, sendo que, atualmente, os maiores são relacionados à segurança. Somado aos problemas clássicos da comunicação sem fio, a topologia dinâmica das MANETs facilita a ação de adversários, tornando-as susceptíveis a diversos tipos de ataques, passivos e ativos (BANERJEE; SWAMINATHAN, 2011; DJENOURI; KHELLADI; BADACHE, 2005).

Em um ataque passivo, um adversário não autorizado tenta descobrir ou utilizar as informações do sistema, mas sem interagir com a rede. Já em um ataque ativo, o adversário tenta invadir um sistema com o objetivo principal de afetar a sua operação (SHIREY, 2000). Em outras palavras, um ataque é considerado passivo quando o atacante apenas monitora ou captura os dados que estão sendo trafegados no sistema, e é considerado ativo quando existe alguma modificação de mensagens ou a criação de dados, afetando o comportamento da rede (ANJUM; MOUCHTARIS, 2007; MICHIARDI; MOLVA, 2003).

Entre as principais particularidades que podem afetar a segurança das MANETs, destacam-se:

a. **ausência de infraestrutura**: estas redes não dependem de nenhuma infraestrutura para suportar as suas operações. Dessa forma, qualquer solução clássica baseada em autoridades certificadoras ou servidores *online* não é aplicável;

b. **segurança física limitada**: as MANETs são vulneráveis às ameaças de segurança física, pois herdam estes problemas das redes sem fio tradicionais, tais como a alta possibilidade de ataques de escuta não-autorizada, falsificação e negação de serviço;

c. **falta de controle centralizado**: as MANETs são redes autônomas e não devem possuir qualquer infraestrutura de administração centralizada. Por isso, a detecção de ataques torna-se difícil, visto que não é fácil monitorar, de forma distribuída, o tráfego de dados em uma rede de larga escala; e

d. **topologia dinâmica**: os nós são livres para se mover arbitrariamente. Dessa forma, a topologia da rede pode mudar aleatória e imprevisivelmente, podendo resultar em frequentes alterações de rotas, particionamento da rede e perdas de dados, o que pode afetar no correto funcionamento dos algoritmos e dificultar a implementação de soluções distribuídas de segurança.

Devido a estas particularidades, o desenvolvimento de aplicações para as MANETs pode ser altamente complexo (ARRUFAT; PARÍS; LóPEZ, 2008). Nas redes em geral, para auxiliar na solução dos problemas de heterogeneidade e distribuição e permitir a

implementação de aplicações mais complexas e flexíveis, são utilizados os serviços de *middleware* (BERNSTEIN, 1996). Recentemente, diversas soluções de *middleware* têm sido propostas para suportar a distribuição das aplicações e serviços nas MANETs. Estas soluções são orientadas a mensagens (HADIM; AL-JAROODI; MOHAMED, 2006b) e são classificadas, neste trabalho, em baseadas em espaço de tuplas, baseadas em *Peer-to-Peer* (P2P), baseadas em contexto e *cross-layer*. Esses *middlewares* são apresentados e comparados no capítulo 2. Os *middlewares* são descritos considerando os serguintes serviços: suporte a grupos, descoberta de recursos, localização de nós e, principalmente, segurança. Segundo (HADIM; AL-JAROODI; MOHAMED, 2006a), tais serviços são importantes nas MANETs devido às características dinâmicas e imprevisíveis destas redes, e portanto são utilizados como parâmetros para a avaliação dos *middlewares*.

Considerando as necessidades de segurança que as MANETs apresentam, e como o *middleware* gerencia toda a comunicação entre um cliente e uma aplicação, ele deve também incluir os aspectos de segurança (AL-JAROODI et al., 2010). No entanto, as soluções de *middleware* disponíveis não consideram, ou consideram apenas parcialmente, os requisitos de segurança das MANETs. Assim, o requisito de segurança pode ser considerado com uma das principais lacunas nas soluções apresentadas. Diante disso, surge a necessidade de se desenvolver uma solução de *middleware* que considere os aspectos de segurança das MANETs e apresente um conjunto de componentes que forneçam serviços seguros às aplicações que utilizam este *middleware*.

Qualquer rede segura deve fornecer cinco propriedades básicas (BANERJEE; SWAMINATHAN, 2011): confidencialidade, disponibilidade, integridade, autenticidade e irretratabilidade. A Tabela 1.1 apresenta as cinco propriedades básicas de um serviço de rede seguro e os seus objetivos.

A segurança pode ser garantida em um *middleware* integrando técnicas de segurança, porém preservando todas as suas funcionalidades essenciais. Posicionar as funcionalidades de segurança no *middleware* logo abaixo das aplicações é interessante por manter a abstração, portabilidade e automação das aplicações, enquanto mantém a transparência da complexidade da rede (AL-JAROODI et al., 2010).

Tabela 1.1: Objetivos das propriedades básicas de segurança

| **Propriedade** | **Objetivo** |
|---|---|
| Confidencialidade | proteção dos dados contra a leitura não autorizada |
| Disponibilidade | recursos ou nós devem estar acessíveis sempre que solicitados |
| Integridade | informações devem ser alteradas apenas por nós autorizados e não podem ser corrompidas |
| Autenticidade | garante a identificação inequívoca de todas as entidades comunicantes |
| Irretratabilidade | emissor ou receptor não podem negar que uma mensagem sua foi transmitida |

## 1.2 Objetivos

Diante das características apresentadas, este trabalho tem como objetivo geral *"propor uma nova solução de middleware seguro para as MANETs, que auxilie nas tomadas de decisão relacionadas a segurança, fornecendo confiabilidade às aplicações"*. Para alcançar este objetivo, alguns objetivos específicos foram determinados:

a. estudar as soluções de *middleware* propostas para as redes *ad hoc* móveis, identificando suas características, vantagens e desvantagens;

b. classificar as soluções de *middleware* existentes;

c. identificar a melhor forma de organização dos nós no novo *middleware*;

d. propor a arquitetura do novo *middleware* seguro para as MANETs;

e. apresentar a integração do módulo de segurança com os demais componentes do *middleware*;

f. detalhar o funcionamento de cada um dos componentes do módulo de segurança.

## 1.3 Contribuições

Este trabalho propõe um novo *middleware* seguro para as MANETs, chamado de Middleware Seguro para a Redes Ad Hoc Móveis (*SEcure Middleware for mobile Ad hoc Networks* (SEMAN)). Este é baseado em grupos, que são formados usando informações de

contexto. Os membros de um grupo trocam informações utilizadas na tomada de decisões e no fornecimento de serviços. Com o uso de uma abordagem baseada em grupos espera-se que nós possam tomar decisões de segurança mais eficazes, visto que os membros de um grupo cooperam entre si no fornecimento dos serviços seguros. O *middleware* possui um módulo de segurança que tem como objetivo garantir a confiabilidade do sistema e a resistência aos seguintes ataques maliciosos: egoísmo, bizantino, personificação e *Sybil*.

Dentre as contribuições deste trabalho, podem ser destacadas:

a. estudo das soluções de *middleware* propostas para redes *hoc* móveis;

b. categorização dessas soluções em cinco categorias: baseadas em espaço de tuplas, baseadas em P2P, baseadas em contexto, *cross-layer*, e orientadas à aplicação;

c. proposta de um novo *middleware* seguro para as redes *ad hoc* móveis;

d. desenvolvimento de um esquema de gerenciamento e avaliação de confiança que considera o contexto das aplicações e pode ser integrado ao SEMAN;

e. desenvolvimento de um esquema de gerenciamento de chaves baseado em identidade totalmente distribuído e integrado ao núcleo criptográfico do SEMAN; e

f. elaboração de um esquema de gerenciamento de grupos baseados em contexto e definição de estratégias para a comunicação segura em grupo.

## 1.4 Organização do trabalho

O restante da tese está organizado da seguinte forma:

**Capítulo 2)** descreve uma visão geral dos *middlewares* e apresenta as soluções de *middleware* desenvolvidas para as MANETs. As soluções apresentadas são classificadas em cinco categorias: baseadas em espaços de tuplas, baseadas em P2P, baseadas em contexto, *cross-layer*, e orientadas a aplicações. Dentro de cada categoria, os *middlewares* são comparados considerando características como: suporte a grupos, descoberta de recursos, localização, e, principalmente, segurança.

**Capítulo 3)** apresenta a nova proposta de *middleware* seguro para as redes *ad hoc* móveis. Neste capítulo é descrito o funcionamento geral do novo *middleware* e como os seus componentes e módulos são integrados a fim de oferecer maior segurança às aplicações.

**Capítulo 4)** detalha o funcionamento do gerenciamento de confiança do módulo de segurança do *middleware* proposto. Este capítulo descreve as operações do esquema de gerenciamento de confiança e como ele fornece as evidências de confiança aos demais componentes do *middleware*.

**Capítulo 5)** descreve detalhadamente o funcionamento do gerenciamento de chaves integrado ao *middleware*. São apresentadas as suas operações e como ele permite o gerenciamento dinâmico do material criptográfico utilizado nas operações de criptografia do *middleware*.

**Capítulo 6)** apresenta o gerenciamento de grupo que é utilizado pelo *middleware* para organizar os nós em grupos de interesse no fornecimento de serviço às aplicações. Ele também descreve como é possível integrar os componentes para fornecer comunicação segura de grupo aos usuários do *middleware*.

**Capítulo 7)** discute alguns cenários nos quais o SEMAN pode ser empregado, com parâmetros distintos de configuração e diferentes restrições de segurança.

**Capítulo 8)** contém as conclusões, contribuições gerais e sugestões de trabalhos futuros.

# CAPÍTULO 2

# MIDDLEWARE: CONCEITOS E VISÃO GERAL

Dentre as muitas definições na literatura, um *middleware* pode ser definido como uma camada de interface sobre o sistema operacional e abaixo das aplicações, como ilustrado na figura 2.1 (BERNSTEIN, 1996). Tipicamente, ele possibilita a interação e a comunicação entre aplicações diferentes por meio de *Application Programming Interfaces*(APIs) através de componentes distribuídos. O objetivo principal de um *middleware* é simplificar os sistemas distribuídos, nos quais os desenvolvedores de aplicações abstraem as implementações das camadas mais baixas. Por fim, ele mascara a heterogeneidade das diversas arquiteturas, sistemas operacionais, linguagens de programação e tecnologias de rede, para facilitar o gerenciamento e o desenvolvimento de aplicações (GEIHS, 2001).

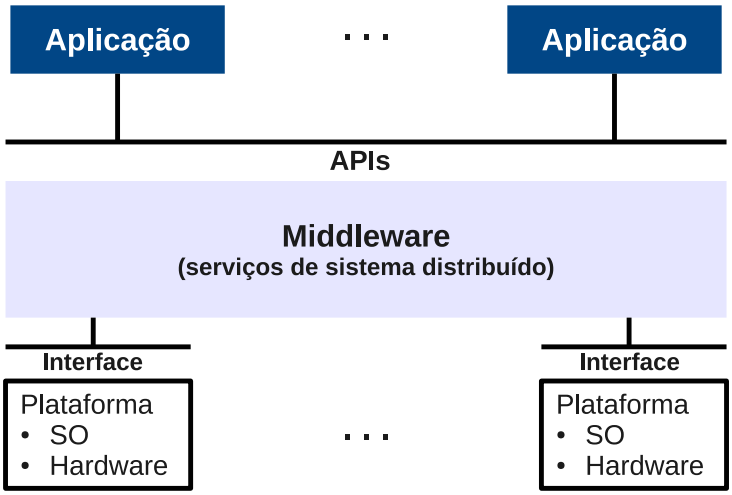



Figura 2.1: Middleware.
Fonte: Adaptado de (BERNSTEIN, 1996)

Na literatura existem quatro tipos clássicos de *middleware* (MASCOLO; CAPRA; EMMERICH, 2002): procedural, objeto/componente, transacional e orientado a mensagens. Essa classificação não é rígida, sendo possível classificar os sistemas de *middleware* usando outras abordagens. Em geral, essas soluções têm como objetivo as redes fixas tradicionais, como a Internet. Dessa forma, nem todas as abordagens são atrativas para

as MANETs, principalmente devido ao alto dinamismo destas redes. Uma característica importante para as redes móveis é o tipo de comunicação utilizado pelo *middleware*.

Nesse contexto, a comunicação pode ser síncrona ou assíncrona. Nos *middlewares* com comunicação síncrona, os participantes permanecem bloqueados durante a comunicação. Já nos *middlewares* com comunicação assíncrona, o requisitante é liberado logo após o envio do pedido de comunicação (TANENBAUM; STEEN, 2007). Assim, a comunicação assíncrona é mais atrativa, por permitir que os nós não estejam todos *online* e acessíveis quando um pedido é realizado ao *middleware*. A Tabela 2.1 apresenta uma breve descrição das soluções clássicas de *middleware*, apresentando, também, se elas suportam comunicação assíncrona.

Tabela 2.1: Visão geral dos tipos tradicionais de middleware

| **Abordagem** | **Comunicação Assíncrona** | **Descrição** |
|---|---|---|
| Procedural | não | clientes invocam serviços dos servidores via chamadas de procedimentos remotas |
| Objeto/ componente | não | suporta a comunicação entre objetos distribuídos, em que um objeto cliente solicita a execução de uma operação de um objeto servidor |
| Transacional | sim | implementa transações, garantindo que as operações necessárias são executadas em todos os nós do sistema |
| Orientado a mensagens | sim | suporta a comunicação entre os componentes distribuídos via trocas de mensagens entre as aplicações |

As soluções de *middleware* desenvolvidas para os sistemas tradicionais não podem ser aplicadas em um ambiente móvel, pois apresentam uma carga computacional pesada e, geralmente, suportam apenas comunicações síncronas (HADIM; AL-JAROODI; MOHAMED, 2006b). Diversos pesquisadores têm projetado soluções de *middleware* para as MANETs. Todas as soluções propostas são orientadas a mensagens. Neste trabalho, esses *middlewares* foram classificados em: baseados em espaço de tuplas, baseados em P2P, baseados em contexto, *cross-layer*, e orientados a aplicação. Nas próximas seções, as soluções de *middleware* para as MANETs são agrupadas de acordo com esta classificação e comparadas entre si, considerando parâmetros como suporte a grupos, descoberta de recursos, localização e segurança. Esses parâmetros foram selecionados por serem importantes no

fornecimento de serviços às aplicações nestas redes.

## 2.1 Baseados em Espaços de Tuplas

O espaço de tuplas é a implementação de uma memória associativa que fornece comunicação assíncrona, anônima e baseada em conteúdo, desacoplando os componentes da aplicação no tempo, espaço e fluxo (GELERNTER, 1985). As informações são organizadas em tuplas e consultadas de forma associativa por meio de mecanismos de buscas por padrões de conteúdo (CABRI; LEONARDI; ZAMBONELLI, 2000). As tuplas contém itens de dados e são escritas ou lidas nos espaços de tuplas. As operações de leitura são realizadas especificando parcialmente o conteúdo das tuplas desejadas.

O *middleware* baseado em espaços de tuplas possibilita às aplicações trocarem dados anonimamente, pois as tuplas são endereçadas de forma associativa, especificando o seu conteúdo. Para suportar as operações usando os espaços de tuplas em redes tradicionais, Gelernter et. al (1985) (GELERNTER, 1985) desenvolveram o modelo Linda, que fornece um conjunto de operadores básicos que podem ser incorporados às linguagens de programação como C, Pascal, Java ou Python. O modelo Linda define quatro operadores básicos para serem executados em um espaço de tuplas global e único:

a. OUT, para inserir uma tupla e disponibilizá-la a todos os processos;

b. IN, para pesquisar e extrair uma tupla;

c. RD, para pesquisar uma tupla e mantê-la disponível aos demais processos; e

d. EVAL, para criar um processo para avaliar tuplas, disponibilizando o resultado ao demais processos.

Em 1998, foi apresentado o L$^2$imbo (DAVIES et al., 1998), um *middleware* alternativo para redes móveis com infraestrutura baseado em espaços de tuplas. O L$^2$imbo é baseado no Linda (GELERNTER, 1985) e inclui extensões que endereçam os requisitos específicos para as operações nas redes móveis. Algumas dessas extensões são: múltiplos espaços de tuplas, hierarquia explícita de tipo de tuplas, tuplas com atributos de Qualidade de

Serviço (*Quality of Service* (QoS)) e componentes de sistema que fornecem serviços para o monitoramento de QoS, a criação de novos espaços de tuplas e a propagação de tuplas entre espaços de tuplas. No L$^2$imbo, os espaços de tuplas são implementados de forma distribuída: cada nó gerencia uma réplica do espaço de tuplas, o que permite operações desconectadas. Contudo, o L$^2$imbo não considera características particulares das MANETs, como a topologia dinâmica e a auto-organização dos nós.

Um outro *middleware* baseado em espaço de tuplas para redes móveis com infraestrutura é o MARS (CABRI; LEONARDI; ZAMBONELLI, 2000), que fornece aplicações mais confiáveis usando agentes móveis, que migram de um nó para outro. Um agente móvel é processo que possui a capacidade de se mover entre os nós de uma rede. Cada nó mantém um espaço de tuplas local que é acessado por agentes que residem nele. O espaço de tuplas permite que um agente responda a ações executadas no espaço de tuplas. No MARS, um agente pode realizar operações de coordenação apenas com outros agentes localizados no mesmo nó, sendo necessária a migração de um agente para a comunicação inter-nó. Isso torna o *middleware* ineficiente, pois requer uma migração para cada operação, que é mais custoso do que uma troca de mensagens. Outros exemplos de *middleware* baseados em espaço de tuplas com agentes móveis são o JMAP (CHUNLIN et al., 2002) e o xSpace (BELLUR; BONDRE, 2006), mas eles não são totalmente aplicáveis para as MANETs.

As próximas seções apresentam os diversos *middlewares* baseados em espaços de tuplas que foram desenvolvidos para as redes *ad hoc* móveis.

### 2.1.1 Linda In a Mobile Environment (LIME) (2001)

O *middleware Linda In a Mobile Environment* (LIME) (MURPHY; PICCO; ROMAN, 2001; MURPHY; PICCO; ROMAN, 2006) é baseado no Linda (GELERNTER, 1985) e fornece uma camada de coordenação que pode ser explorada para suportar o desenvolvimento de aplicações com mobilidade lógica e/ou física. Contudo, enquanto no Linda o contexto para a computação é representado por um espaço de tuplas persistente e global, no LIME esse contexto é disponibilizado por meio de um compartilhamento dinâmico

de espaços de tuplas, baseado na conectividade entre os nós. Cada espaço de tuplas é associado permanentemente a um nó e possui regras para o compartilhamento dinâmico, ou transiente, baseadas no conteúdo das tuplas e na localização física dos nós.

Os espaços de tuplas individuais dos nós são chamados de *Interface Tuple Space* (ITS) e contém as tuplas que o nó está disponibilizando e as tuplas co-localizadas com ele. Além disso, os ITSs podem ser compartilhados entre os nós que formam um grupo. Quando múltiplos agentes móveis, em um mesmo nó ou não, são capazes de se comunicar, direta ou transitivamente, eles formam um grupo. Os conteúdos dos ITSs de todos os membros de um grupo são unidos, ou compartilhados, para formarem um único grande contexto que é acessado por um agente via seu próprio ITS. O compartilhamento é transparente para cada usuário.

A figura 2.2 ilustra a estrutura do LIME. Os nós móveis contém agentes móveis, que são os únicos componentes que podem transportar um ITS. Dois nós estão conectados quando a distância entre eles é menor que o raio de alcance da comunicação de suas antenas. Já dois agentes móveis estão conectados apenas quando eles estão co-localizados em um mesmo nó ou estão em nós interconectados.

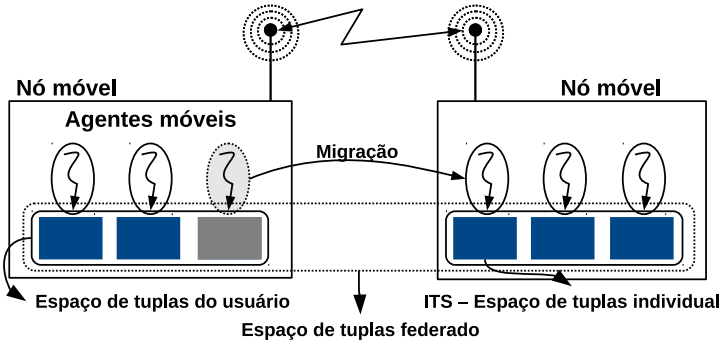



Figura 2.2: O modelo do LIME.
Fonte: Adaptado de (MURPHY; PICCO; ROMAN, 2006)

A associação dos ITSs dos agentes móveis de um nó formam o espaço de tuplas do usuário. Os espaços de tuplas de usuários de nós interconectados podem ser associados e formam um espaço de tuplas federado. Quando um agente móvel realiza uma consulta em seu ITS, o *middleware* retorna, transparentemente, uma tupla de qualquer ITS que pertence ao espaço de tuplas federado.

### 2.1.2 Tuples On The Air (TOTA) (2003)

O *Tuples On The Air* (TOTA) (MAMEI; ZAMBONELLI; LEONARDI, 2003) foi projetado para garantir a computação distribuída em redes dinâmicas, incluindo as MANETs. Ele fornece interações desacopladas, adaptativas e cientes do contexto. O TOTA depende de tuplas distribuídas no espaço, que não estão associadas com nós específicos mas são injetadas na rede e propagadas de acordo com padrões específicos das aplicações.

Cada nó TOTA gerencia referências para um conjunto limitado de nós vizinhos, que nas MANETs depende do raio de comunicação dos nós. Esses nós são capazes de armazenar tuplas e disseminá-las pela rede, seguindo uma regra de conteúdo e propagação. No TOTA, as tuplas não são necessariamente réplicas distribuídas e podem ser usadas para construir uma estrutura de dados sobreposta, distribuída, que expressa algum tipo de informação contextual e espacial. Por exemplo, em cenários de gerenciamento de tráfego, as informações sobre o estado de um semáforo são relevantes apenas para os carros próximos deste semáforo. Dessa forma, as tuplas contendo tal informação (regra de conteúdo) deveriam ser propagadas apenas a 300 metros de distância do semáforo (regra de propagação).

A figura 2.3 ilustra a arquitetura do TOTA, composta por três partes: a API TOTA, o Motor TOTA e a Interface de Eventos. A API TOTA é uma interface entre a aplicação e o *middleware* que fornece mecanismos para as aplicações injetarem novas tuplas, acessarem o espaço de tuplas local e colocarem as assinaturas na Interface de Eventos. O Motor TOTA é o "núcleo" do *middleware*, responsável pela manutenção da rede TOTA, gerenciamento da propagação das tuplas, monitoramento e reconfiguração da rede, injeção de novas tuplas e re-propagação das tuplas armazenadas. Finalmente, a Interface de Eventos notifica a aplicação sobre a chegada de novas tuplas ou sobre a saída e entrada de nós vizinhos.

### 2.1.3 Limone (2004)

O Limone (FOK; ROMAN; HACKMANN, 2004) tem como objetivo facilitar o desenvolvimento de aplicações sobre as MANET com agentes e nós móveis. Ele tem a única premissa de que a troca de mensagens é possível e oferece um conjunto de garantias funcio-

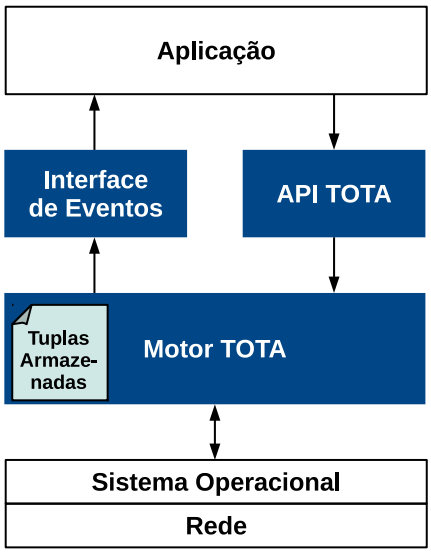



Figura 2.3: A arquitetura do TOTA.
Fonte: Adaptado de (MAMEI; ZAMBONELLI; LEONARDI, 2003)

nais. Nele, cada agente tem uma lista de conhecimento que contém uma visão dos agentes remotos na sua proximidade. Para cada agente, o Limone descobre agentes remotos e atualiza sua lista de conhecimento de acordo com políticas personalizadas. Da perspectiva das aplicações, todas as interações com outros componentes ocorrem referenciando os membros da lista de conhecimento.

A arquitetura do Limone é representada na Figura 2.4. O Limone fornece um ambiente de execução chamado de Servidor Limone, que atua entre os agentes e o sistema operacional. Uma aplicação usa o Limone interagindo com um agente, que contém o espaço de tuplas, a lista de conhecimento, o registro de reações, a lista de reações e um gerente de operações.

Um agente permite que a aplicação personalize os seguintes componentes: perfil, política de acoplamento e gerente de operações. O perfil é o conjunto de objetos que descreve as propriedades de um agente. A política de acoplamento especifica quais agentes são relevantes baseados em seus perfis. Finalmente, o gerente de operações identifica quais pedidos de operações remotas são aceitos.

Algumas operações fornecidas pelo Limone são:

a. mecanismo de descoberta: responsável pela descoberta da entrada e saída de outros agentes;

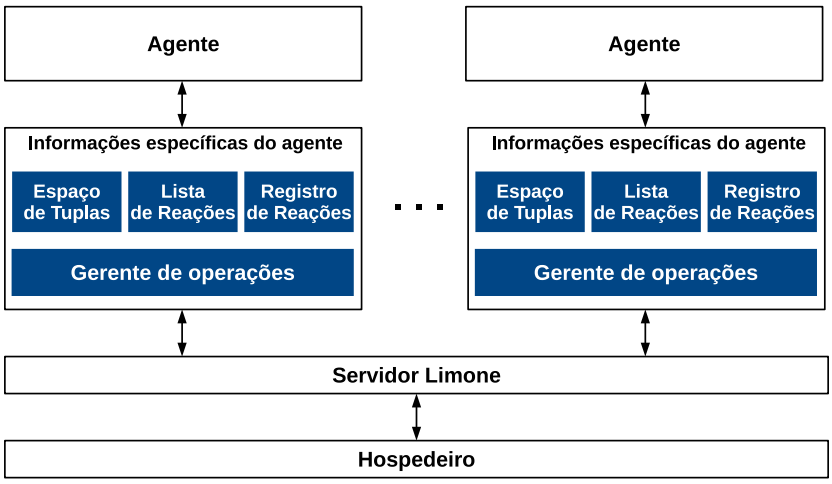



Figura 2.4: A arquitetura do Limone.
Fonte: Adaptado de (FOK; ROMAN; HACKMANN, 2004)

b. gerenciamento do espaço de tuplas: todo dado de aplicação é armazenado em espaços de tuplas individuais, gerenciados pelo *middleware*;

c. mecanismo de reação: permite a um agente informar outros agentes dentro de sua lista de conhecimento que ele está interessado nas tuplas que combinam com um padrão específico; e

d. mobilidade do agente: suporta mobilidade leve preservando código e estado.

### 2.1.4 Coordination Across Space & Time (CAST) (2006)

O *Coordination Across Space & Time* (CAST) (ROMAN; HANDOREAN; SEN, 2006) é projetado para suportar a comunicação entre agentes em execução em nós distintos. Para endereçar algumas características particulares das MANET, ele considera o uso de um algoritmo de roteamento desconectado, um tipo de roteamento que não garante a conectividade fim-a-fim entre a origem e o destino. Segundo os autores, esse algoritmo seria similar ao *Dynamic Source Routing* (DSR)[1], pois a sequência completa de nós intermediários seria definida pela origem. Contudo, ele difere na maneira na qual os nós que formam essa rota são selecionados. Os autores do CAST sugerem que as rotas sejam criadas usando informações sobre a mobilidade dos nós para calcular intervalos de conec-

[1] Uma breve explicação do DSR é apresentada no Apêndice A.2

tividade entre eles. Tais informações são trocadas entre os nós, que propagam seu perfil de movimento, como um plano de mobilidade, pela rede.

O CAST é um *middleware* de coordenação que fornece técnicas para integrar as funcionalidades e serviços de aplicações heterogêneas. No CAST, cada espaço de tuplas é associado a um único nó. O escopo da coordenação é controlado de forma que nós remotos possam ser alcançados em qualquer tempo e espaço. A coordenação no tempo é associada com a capacidade de especificar um tempo de vida para os dados e as operações. A coordenação no espaço está relacionada à capacidade de identificar uma localização ou área onde as operações ou dados estão alocados.

Embora os autores argumentem que usam espaço de tuplas para garantir as operações de coordenação e gerenciar as operações espaço-temporais, eles não descrevem como usar esses espaços de tuplas. A principal contribuição do CAST é o uso do conceito do algoritmo de roteamento desconectado para aumentar a efetividade do *middleware*.

### 2.1.5 MESH*Mdl* (2007)

O MESH*Mdl* (HERRMANN; MüHL; JAEGER, 2007) fornece um alto nível de ciência de contexto e desacoplamento dos componentes das aplicações. O meio de comunicação central é o *Espaço de Eventos*, baseado nos espaços de tuplas. O uso do Espaço de Eventos permite que o *middleware* submeta informações atualizadas sobre o seu contexto atual às aplicações, deixando-as cientes da dinâmica da rede. Os componentes MESH*Mdl* são altamente autônomos e desacoplados, capazes de se adaptar e reagir a alterações no ambiente.

A figura 2.5 ilustra a arquitetura do MESH*Mdl*, composta de quatro subsistemas centrais: Espaço de Eventos, Agente de Tempo de Execução, Gerente de Interações e Camada de Conexão Genérica. O Espaço de Eventos está no centro do *middleware* e fornece a funcionalidade básica do espaço de tuplas. O Agente de Tempo de Execução é responsável pela execução e manutenção dos agentes de aplicação. O Gerente de Interações é responsável por qualquer comunicação com os nós vizinhos. Finalmente, a Camada de Conexão Genérica é uma rede *ad hoc* abstrata, que oferece uma API ao Gerente de

Interações para descobrir dispositivos e para configurar a conexão com os nós vizinhos.

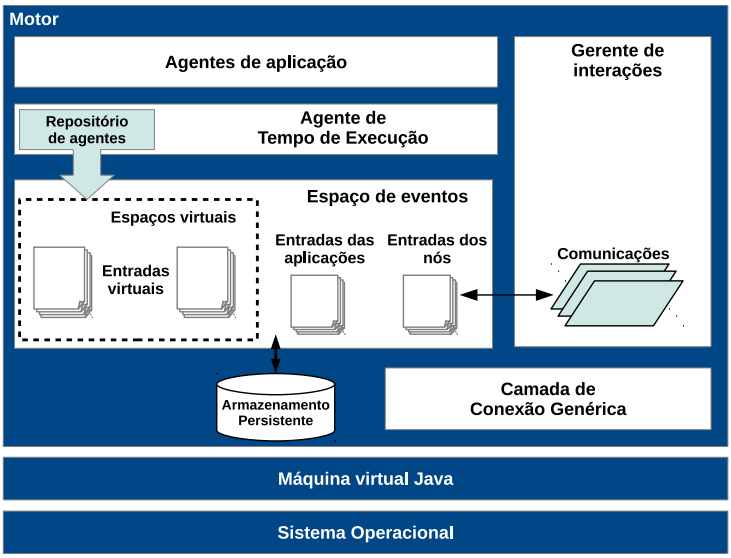



Figura 2.5: A arquitetura do MESH*Mdl*.
Fonte: Adaptado de (HERRMANN; MüHL; JAEGER, 2007)

No MESH*Mdl* as notificações de eventos e comunicações inter-agentes são publicadas via Espaço de Eventos. Nele, as tuplas são chamadas de *Entries* (Entradas) e podem ser implementadas como objetos sem métodos. Os atributos de cada Entrada são usados para as comparações baseadas em contexto nas operações de leitura. Qualquer tupla sinalizada como persistente é armazenada em um diretório dedicado, chamado de Armazenamento Persistente. Quando um agente deseja disponibilizar tuplas aos dispositivos vizinhos, ele deve executar uma escrita remota no Espaço de Eventos dos outros nós.

O MESH*Mdl* também introduz o conceito de Espaços Virtuais, que são interfaces para um módulo MESH*Mdl* se conectar ao Espaço de Eventos. Usando Espaços Virtuais a funcionalidade do *middleware* é aumentada, permitindo módulos funcionais adicionais enquanto mantém uma única interface com o Espaço de Eventos.

### 2.1.6 Comparativo dos middleware baseados em espaços de tuplas

A tabela 2.2 apresenta uma comparação das soluções de *middleware* baseadas em espaço de tuplas. Embora o espaço de tuplas pareça ser atrativo para as MANETs devido

às operações desacopladas, requisitos importantes são ignorados nas soluções propostas, tais como o suporte a grupos. Com exceção do LIME, nenhum *middleware* baseado em espaço de tuplas fornece suporte a grupos. Além disso, mesmo o LIME fornece apenas um suporte restrito, limitado ao conteúdo e à vizinhança dos nós.

Tabela 2.2: Comparativo dos *middleware* baseados em espaço de tuplas

| **Middleware** | **Suporte a grupos** | **Descoberta de recursos** | **Localização** | **Segurança** |
|---|---|---|---|---|
| LIME | pelo conteúdo e limitado à vizinhança | limitado às notificações | sugere o uso de mecanismos de posicionamento, como GPS | ND |
| TOTA | ND | limitado às notificações | ND | ND |
| Limone | ND | difusão periódica de *beacons* | ND | ND |
| CAST | ND | ND | baseado em fofocas | ND |
| MESH*Mdl* | ND | informações trocadas em encontros | ND | anonimato |

O serviço de descoberta de recursos é o melhor suportado pelos *middlwares* baseados em espaços de tuplas. Apenas o CAST não menciona qualquer forma de realizar a descoberta de recursos. O LIME e o TOTA usam mensagens de notificação e o Limone usa difusão periódica de *beacons*, sendo que as duas abordagens podem gerar um alto custo de comunicação ao sistema. Já o MESH*Mdl* realiza trocas de informações nos encontros físicos, que é mais atrativo para as MANETs, pois a mobilidade dos nós possibilita a disseminação rápida das informações.

Por fim, as soluções de *middleware* baseadas em espaços de tuplas parecem ignorar a importância do serviço de localização e a segurança para as MANETs. A localização é apenas mencionada no LIME, que sugere o uso de mecanismos de posicionamento. Já a segurança nem chega a ser mencionada e somente o MESH*Mdl* fornece uma forma de anonimato, que é uma pequena parte de um serviço de segurança.

## 2.2 Baseados em P2P

A computação P2P está se tornando um paradigma comum para muitas aplicações distribuídas, permitindo o compartilhamento de recursos e a comunicação direta entre

pares, ou nós. Várias soluções de *middleware* (BISIGNANO et al., 2003; BISIGNANO et al., 2004; BISIGNANO; MODICA; TOMARCHIO, 2005; KORTUEM, 2002; WANG; BJORNSGARD; SAXLUND, 2007) para as MANETs têm usado as abordagens dos *middleware* para redes P2P, principalmente devido à característica descentralizada destas redes.

### 2.2.1 Proem (2002)

O Proem (KORTUEM, 2002) foi desenvolvido pelo *Wearable Computing Laboratory* na Universidade de Oregon (EUA). Ele fornece uma solução para o desenvolvimento e a implantação de aplicações P2P nas MANETs. Seus objetivos incluem suporte ao desenvolvimento em alto nível, independência de plataforma, interoperabilidade e extensibilidade.

A figura 2.6 ilustra a arquitetura do *middleware*. As aplicações, chamadas de *peerlets*, utilizam o modelo de programação baseada em eventos. Tais eventos são acionados nas seguintes situações: (*i*) como reação às alterações no seu estado interno e no contexto externo; (*ii*) como reação às mensagens recebidas dos nós próximos.

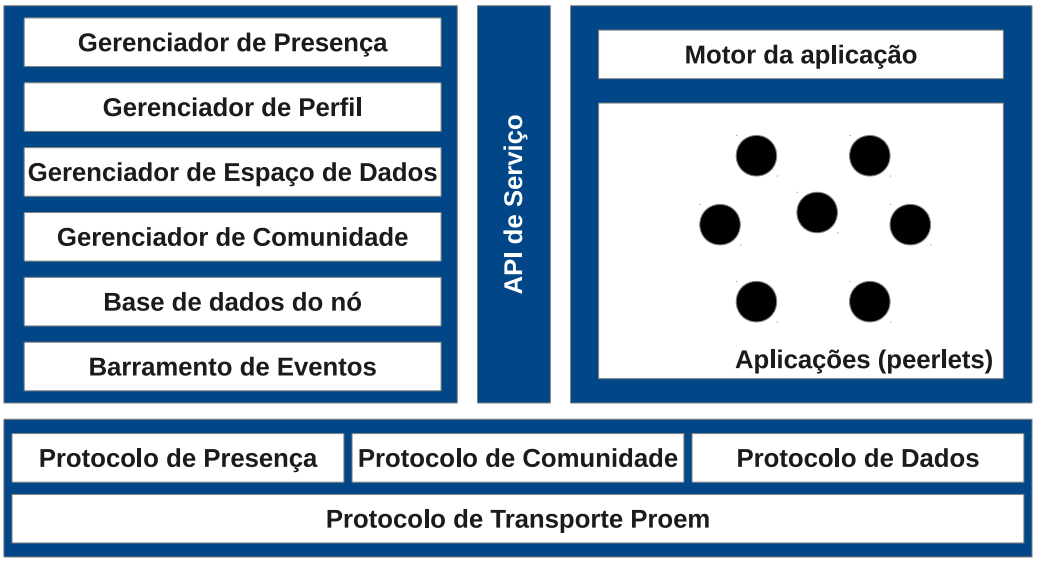



Figura 2.6: Arquitetura do Proem.
Fonte: Adaptado de (KORTUEM, 2002)

O núcleo do Proem fornece quatro protocolos de comunicação que definem a sintaxe e a semântica das mensagens trocadas pelos nós. O Protocolo de Transporte Proem é assíncrono, sem conexão e não confiável, dando suporte à comunicação básica entre os

nós. O Protocolo de Presença permite que os nós anunciem a sua presença e descubram a presença de outros nós no sistema. O Protocolo de Dados possibilita o compartilhamento e a sincronização dos dados. Finalmente, o Protocolo de Comunidade é responsável pelo estabelecimento das relações de confiança entre os nós e formação de grupos.

O Proem fornece seis serviços às aplicações que são executadas como um conjunto de APIs de alto nível. O gerenciador de presença é responsável pela descoberta de nós próximos. O gerenciador de perfil mantém as informações sobre a identidade dos nós e os recursos compartilhados. O gerenciador de espaços de dados realiza o armazenamento persistente dos espaços de dados e controla o acesso a esses dados. O gerenciador de comunidade registra a associação do nó em grupos e valida outras associações de grupo. O banco de dados do nó mantém um registro persistente dos encontros com outros nós, permitindo que as aplicações determinem quando e com que frequência um nó em particular foi encontrado no passado. Finalmente, o barramento de eventos possibilita a comunicação baseada em eventos entre as aplicações.

### 2.2.2 ExPeerience (2003)

O ExPeerience (BISIGNANO et al., 2003; BISIGNANO et al., 2004) fornece uma camada capaz de esconder a complexidade das MANETs para os programadores, permitindo o desenvolvimento de aplicações que podem usar as peculiaridades de tais ambientes. Ele utiliza os serviços fornecidos por um ambiente P2P. Para isso, ele utiliza um *framework* P2P, chamado JXTA (JXTA, 2014), que fornece interoperabilidade, independência de plataforma e ubiquidade.

Segundo os autores, embora o JXTA seja altamente eficaz, ele não endereça algumas características chaves das MANETs, como, por exemplo, o gerenciamento de conexões intermitentes. O objetivo do ExPeerience é aumentar alguns serviços do JXTA, fornecendo uma outra camada que considera as características das MANETs, mantendo alta compatibilidade com as redes JXTA tradicionais. A figura 2.7 ilustra a arquitetura do *middleware*. A camada Motor, construída sobre o JXTA, fornece uma interface às aplicações. Algumas funcionalidades introduzidas pelo ExPeerience são: gerenciamento de

conexões intermitentes e múltiplas interfaces, mecanismos de descoberta de recursos mais eficiente e mobilidade de código.

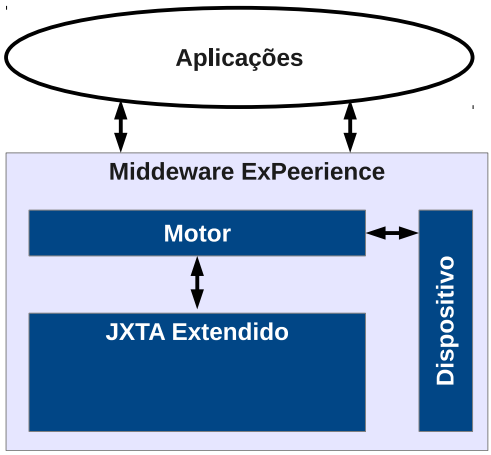



Figura 2.7: Arquitetura do ExPeerience.
Fonte: Adaptado de (BISIGNANO et al., 2003)

No ExPeerience, a estrutura do JXTA foi estendida com o objetivo de tratar múltiplas interfaces. Assim, ele permite o uso de múltiplas interfaces de rede para cada nó e a associação de mais de um endereço para a mesma interface. O serviço TCPTransport do JXTA foi modificado para tratar as conexões intermitentes das MANETs, nas quais os nós podem entrar e sair da rede com frequência.

O serviço de descoberta de recursos fornece uma memória central para tratar as frequentes desconexões e reconexões dos nós. Ele também fornece informações atualizadas sobre os nós e seus serviços compartilhados, baseado em avisos com informações que os nós desejam compartilhar. Todos os avisos possuem um tempo de vida e são mantidos pelo gerenciador de *cache*. Após a expiração do seu tempo de vida, os avisos são excluídos.

O serviço de código móvel define os métodos para o gerenciamento da mobilidade do código. Qualquer nó no ExPeerience é capaz de migrar um serviço de/para outro nó. Pelo servidor de código móvel, os nós são capazes de compartilhar dados e códigos. Essa abordagem possibilita a instalação de novos serviços dinamicamente, permitindo que o *middleware* se adapte a situações imprevisíveis.

### 2.2.3 JMobiPeer (2004)

O *middleware* JMobiPeer (BISIGNANO; MODICA; TOMARCHIO, 2005) também fornece uma camada sobre o JXTA para esconder a complexidade das MANETs no desenvolvimento das aplicações. Ele é uma melhoria do ExPeerience (BISIGNANO et al., 2003; BISIGNANO et al., 2004), sendo que ambos possuem conceitos similares e apresentam funcionalidades em comum.

A arquitetura do JMobiPeer é modular, a fim de torná-lo aplicável e adaptável para qualquer dispositivo. Seus princípios podem ser resumidos nos seguintes pontos: ser compatível com os protocolos JXTA; ser capaz de trabalhar nas MANETs, mesmo quando desconectados das redes JXTA tradicionais; rodar em dispositivos com recursos limitados; superar as limitações da arquitetura *JXTA for Micro Edition* (JXME), um subprojeto do JXTA para dispositivos compatíveis com o *Java 2 Mobile Environment* (J2ME) (J2ME, 2014). Entre as limitações do JXME está a necessidade de um *proxy* para efetuar a comunicação entre os nós.

A figura 2.8 mostra uma visão geral da arquitetura do JMobiPeer, em que todas as camadas são compatíveis com o J2ME. O *middleware* possui duas camadas: serviços e núcleo, que são acessadas pelas aplicações, como mensagens instantâneas. A camada de serviço implementa as funcionalidades para indexar e descobrir recursos. A camada núcleo fornece o Mensageiro Virtual, que suporta a comunicação central, e os módulos Gerenciamento do Nó, Gerenciamento de Grupo, Gerenciamento de Avisos e Descoberta.

O Mensageiro Virtual provê os protocolos de transporte e serviço para gerenciar a comunicação dos nós com a rede. Ele é responsável por abstrair os endereços físicos dos nós na rede lógica e gerenciar a transmissão e recepção das mensagens. O Gerenciamento do Nó mantém as informações relevantes do nó, como endereço, identificador e descrição. O Gerenciamento de Grupo permite iniciar os serviços e os protocolos que podem ser usados pelos nós e mantém a lista dos grupos que o nó pertence. O Gerenciamento de Avisos mantém todos os avisos, que fornecem informações sobre os serviços, nós, grupos e endereços disponíveis. Dessa forma, encontrar nós e todos os seus recursos compartilhados se reduz a uma consulta aos avisos que foram trocados pelos nós. O módulo Descoberta

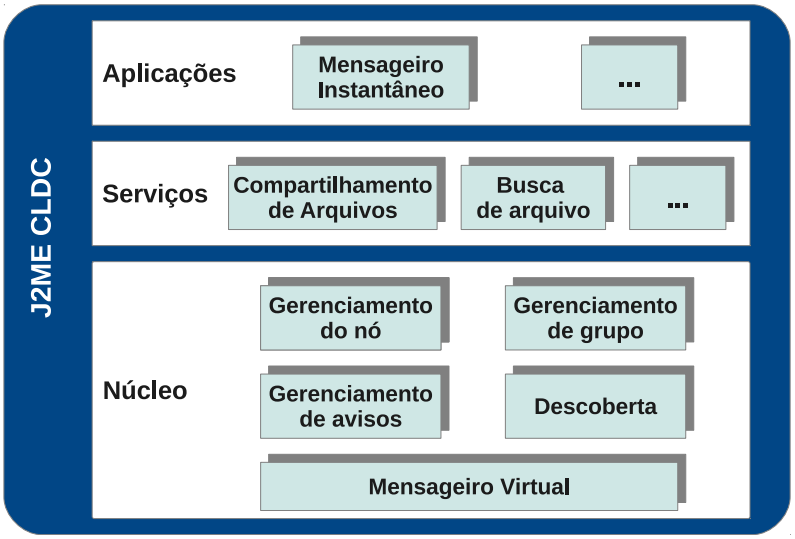



Figura 2.8: Arquitetura do JMobiPeer.
Fonte: Adaptado de (BISIGNANO; MODICA; TOMARCHIO, 2005)

é responsável pela busca e publicação dos avisos para os membros de um grupo.

### 2.2.4 Peer2Me (2007)

O Peer2Me (WANG; BJORNSGARD; SAXLUND, 2007) fornece um *framework* de programação de alto nível e transparente que esconde a tecnologia de rede usada na comunicação, possibilitando o desenvolvimento rápido de aplicações P2P nas MANETs. Ele oferece serviços de descoberta de nós e mensagens para facilitar o desenvolvimento de aplicações colaborativas. Contudo, o Peer2Me é projetado apenas para ser implantado em nós móveis que usam dispositivos Bluetooth.

O Peer2Me considera o uso de J2ME com o *Connected Limited Device Configuration* (CLDC) e o *Mobile Information Device Profile* (MIDP). A figura 2.9 mostra a arquitetura geral do Peer2Me e como ela se encaixa no ambiente J2ME. Além do MIDP e do CLDC do J2ME, o *middleware* usa duas APIs opcionais do J2ME: JSR82 para acessar e gerenciar redes Bluetooth e JSR75 para acessar o Gerenciador de Informações Pessoais.

Como o Peer2Me é desenvolvido para rodar sobre dispositivos Bluetooth, as comunicações devem usar um protocolo mestre-escravo. Por outro lado, o nó mestre pode ser um ponto único de falhas, o que não é desejável nas MANETs. Para mitigar esse problema, as conexões mestre-escravo são estabelecidas dinamicamente quando dois nós desejam se

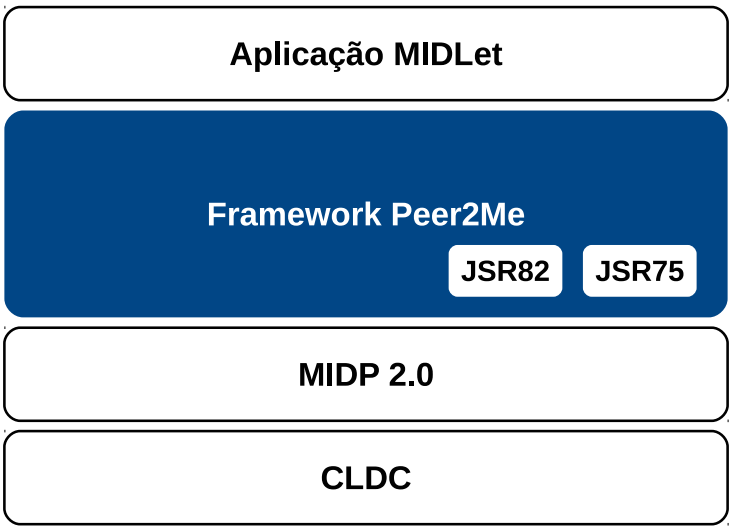



Figura 2.9: Arquitetura do Peer2Me.
Fonte: Adaptado de (WANG; BJORNSGARD; SAXLUND, 2007)

comunicar. Dessa forma, todos os nós se conhecem mutuamente e todos os nós têm a mesma responsabilidade.

A descoberta de novos nós foi implementada usando o protocolo de descoberta Bluetooth fornecido no J2ME que pesquisa todos os dispositivos Bluetooth na proximidade. No Peer2Me, o *middleware* filtra e realiza uma busca por todos os nós rodando um serviço Peer2Me. Após essa busca, o nó compartilha o resultado com todos os nós que ele encontrou. Um busca por novos nós é iniciada quando uma nova aplicação é executada.

### 2.2.5 Comparativos dos middleware baseados em P2P

A tabela 2.3 resume as soluções de *middleware* baseadas em P2P, considerando os requisitos das MANETs. As soluções apresentadas não apresentam nem mecanismos de localização nem de segurança para suportar as operações das MANETs. Por outro lado, todos os *middleware* baseados em P2P fornecem mecanismos para suporte a grupos, já que a organização baseada em grupos é comum nas redes P2P.

Além disso, todas as soluções apresentadas fornecem um meio para a descoberta de recursos. No Proem, os nós anunciam a sua presença e os recursos que eles compartilham. O Peer2Me usa o protocolo de descoberta Bluetooth, não sendo aplicável para as redes de grande escala. Por fim, o ExPeerience e o JMobiPeer usam avisos para informar os

Tabela 2.3: Comparativos dos *middleware* baseados em P2P

| **Middleware** | **Suporte a grupos** | **Descoberta de recursos** | **Localização** | **Segurança** |
|---|---|---|---|---|
| Proem | comunidades baseadas em interesses comuns | Nós anunciam sua presença | ND | ND |
| ExPeerience | sim, baseado em JXTA | via avisos | ND | ND |
| JMobiPeer | sim, baseado em JXTA | via avisos | ND | ND |
| Peer2Me | sim | usando protocolo de descoberta Bluetooth | ND | ND |

nós sobre os recursos disponíveis na rede. Esta estratégia pode gerar um alto custo de comunicação, se todos os nós enviarem avisos sobre todos os recursos que eles conhecem no sistema.

## 2.3 Baseado em contexto

Um contexto pode ser definido para incluir aspectos que podem afetar uma entidade particular (MASCOLO et al., 2002). Nas MANETs, um contexto pode ser um conjunto de nós e suas propriedades que sejam de interesse de um outro nó. Por exemplo, o contexto de um nó $x$ pode ser o conjunto dos nós que podem afetar o seu comportamento, que podem se comunicar com ele ou transportar atividades em seu nome (FREY; ROMAN, 2007). Essa noção de contexto é atrativa nas MANETs porque na ausência de uma infraestrutura fixa ou servidores fixos, todas as informações podem ser associadas com um ou mais nós móveis.

### 2.3.1 Scalable Timed Events And Mobility (STEAM) (2003)

O *Scalable Timed Events And Mobility* (STEAM) (MEIER; CAHILL, 2002; MEIER; CAHILL, 2003) foi projetado para fornecer comunicação entre nós em cenários como gerenciamento de tráfego. Tais cenários têm um grande número de nós desde objetos móveis, como carros e ambulâncias, até os fixos, como sinais de tráfego e radares. No STEAM, esses nós interagem usando comunicação baseada em eventos, trocando informações da situação de tráfego atual.

O STEAM considera que nas aplicações de gerenciamento de tráfego, ou similares, os nós podem querer interagir apenas com os nós próximos. Além disso, ele assume que em tais aplicações as formas mais indicadas para as comunicações entre os nós é via notificações de eventos, ou simplesmente eventos. Ele emprega um modelo publicar/assinar que permite que nós consumidores (receptores) assinem tipos de eventos em particular, sem depender de serviços do sistema ou mediadores.

O *middleware* usa comunicação em grupo para efetuar a comunicação baseada em eventos, argumentando que esta abordagem é mais aplicável para os modelos de comunicação orientada a mensagens (BANAVAR et al., 1999). Também, com base nessa suposição, o serviço de comunicação em grupo é baseado em proximidade. Os grupos de proximidade geográfica e funcional permitem aos componentes da aplicação móvel descobrirem uns aos outros usando *beacons*. Os aspectos geográficos especificam a área onde a informação é válida enquanto os aspectos funcionais o interesse comum dos nós, como as informações de um semáforo, por exemplo.

Como o número de eventos propagados em um sistema baseado em eventos pode ser muito grande e qualquer consumidor pode estar apenas interessado em um subconjunto de eventos, o STEAM fornece o uso de filtros de eventos. Três tipos de filtros de eventos são permitidos: **assunto**, **proximidade** e **conteúdo**. Os filtros por assunto e proximidade são aplicados nos produtores e os eventos filtrados são roteados para os assinantes. Já os filtros por conteúdo são utilizados quando um evento é recebido por um assinante para determinar se o evento deve ser entregue à aplicação ou não.

### 2.3.2 Self-organized Marketplace-based Middleware for MANETs (2004)

No *Self-organized Marketplace-based Middleware for MANETs* (SELMA) (GöRGEN et al., 2004), um contexto é chamado de mercado, uma área geográfica limitada em que a probabilidade de encontrar as informações necessárias é alta. Os autores sugerem que existem várias aplicações como quadros eletrônicos, serviços de informações públicas e leilões *online* que podem usar esse tipo de comunicação. No SELMA os agentes enviam

dados apenas para outros agentes localizados dentro do mesmo mercado. Os agentes, aplicações ou serviços, podem se mover para alvos geográficos diferentes "saltando" de um nó para outro.

Algoritmos distribuídos aproximam o número atual de nós numa área, a fim de formar os mercados. Para isso, ele divide uma área geograficamente limitada em pequenos retângulos. Em seguida, um mecanismo de detecção possibilita aos dispositivos decidir onde posicionar um novo mercado: em geral, uma região retangular com o maior número de nós móveis.

Como ilustrado na figura 2.10, a arquitetura do SELMA é dividida em três partes: abstração da comunicação, plataforma agente e agentes de aplicação e serviço. A abstração da comunicação fornece métodos genéricos para o posicionamento, comunicação sem fio e descoberta de vizinhos. A plataforma agente representa a maior parte das funcionalidades do *middleware*, incluindo os protocolos de roteamento, localização e gerenciamento de mercados. Por fim, a camada mais alta reagrupa a especificação dos dois tipos de agentes: de aplicação e de serviço.

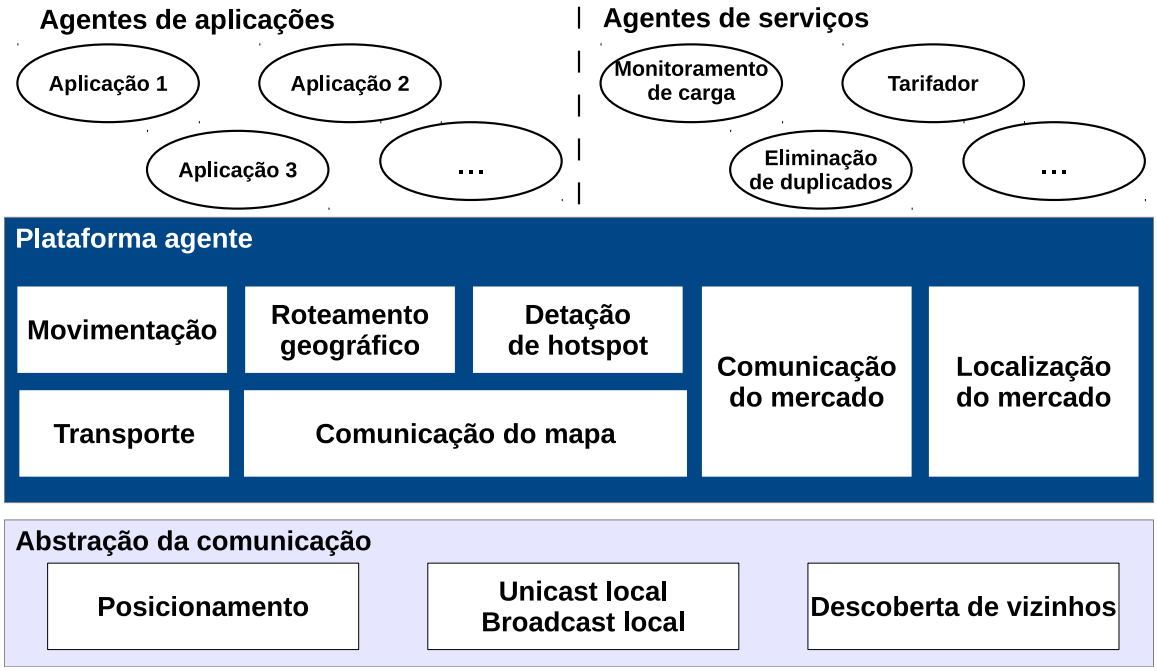



Figura 2.10: A arquitetura do SELMA.
Fonte: Adaptado de (GöRGEN et al., 2004)

O envio e recebimento de dados e agentes entre dispositivos interessados e os mercados

é obtido por meio de variações nos algoritmos de distribuição epidêmica de mensagens e roteamento geográfico. Dois tipos de comunicação são fornecidos: difusão em nível de mercado e endereçamento *unicast* de um agente. Como os agentes podem alterar seus dispositivos de hospedagem durante a comunicação, todas as comunicações são realizadas entre os agente e não entre os nós. As difusões em nível de mercado são geograficamente limitadas, enquanto as mensagens de *unicast* usam roteamento baseado em topologia entre os pares comunicantes.

### 2.3.3 Epidemic Messaging Middleware for Ad hoc networks (2005)

Em (MUSOLESI; MASCOLO; HAILES, 2005), os autores apresentam uma adaptação do *Java Message Service* (JMS) (HAPNER et al., 2002) para ambientes *ad hoc* móveis, chamado de *Epidemic Messaging Middleware for Ad hoc networks* (EMMA). O JMS é uma coleção de interfaces para comunicação assíncrona entre componentes distribuídos. Ele fornece uma forma comum para os desenvolvedores Java criarem, enviarem e receberem mensagens. O EMMA permite a interoperabilidade entre as infraestruturas com fio e *ad hoc*.

O EMMA adapta o JMS para as MANETs alterando o procedimento de troca de mensagens utilizado no JMS e adicionando um mecanismo de roteamento epidêmico para facilitar a entrega de mensagens no ambiente dinâmico. Como no JMS, o EMMA permite o uso de comunicações ponto-a-ponto e/ou publicar/assinar. Na comunicação ponto-a-ponto, as aplicações usam filam para a troca de mensagens assíncronas entre as partes. A localização das filas é determinada pelos requisitos das aplicações, tornado-as ciente de contexto. Para entregar mensagens para os nós fora do raio de transmissão do emissor da mensagem, o protocolo de roteamento epidêmico assíncrono é utilizado. Contudo, o uso de protocolos epidêmicos pode impor limitações de escalabilidade ao *middleware*.

No modelo publicar/assinar, alguns nós contém tópicos e outros nós podem assiná-los. Os tópicos são trocados pelos membros assinantes de um grupo usando um protocolo síncrono ou um protocolo epidêmico. Esse modelo também fornece mecanismos para

manter e remover as assinaturas das mensagens.

### 2.3.4 Allocation and Group Aware Pervasive Environment (2005)

O *Allocation and Group Aware Pervasive Environment* (AGAPE) (BOTTAZZI; CORRADI; MONTANARI, 2005) é um *middleware* para gerenciamento de grupo que explora a visibilidade da informação de contexto para criar e descobrir grupos de interesse, monitorar a disponibilidade dos membros dos grupos e organizar dinamicamente os membros dos grupos. Ele fornece um conjunto de serviços básicos para compartilhar recursos nas MANETs e construir sobre essas redes vários mecanismos de compartilhamento de recursos.

O AGAPE permite que nós vizinhos criem e participem dinamicamente de grupos de interesse sob demanda e fornece todas as facilidades necessárias para propagar ao nível das aplicações a visibilidade dos nós co-localizados, juntamente com os seus perfis e os recursos que eles compartilham. Além disso, apenas os nós com maior capacidade de recursos podem realizar operações de gerenciamento de grupo, enquanto os nós com recursos limitados apenas participam de um grupo para compartilhar recursos.

A figura 2.11 ilustra a arquitetura AGAPE, que é organizada em duas camadas lógicas sobre a *Java Virtual Machine* (JVM): a Camada de Serviços Básicos e a Camada de Gerenciamento de Grupo. A primeira inclui um conjunto de serviços para realizar nomeação, descoberta e monitoramento da disponibilidade dos membros de um grupo. A segunda fornece o suporte necessário para descobrir, criar e excluir os grupos.

A camada de Serviços Básicos compreende três serviços: o Gerenciamento de Rede permite que os nós troquem mensagens pela rede; o Serviço de Proximidade permite que membros de um grupo informem sua disponibilidade em uma localidade propagando avisos em tempos regulares; o Serviço de Nomeação por Proximidade gera aleatoriamente identificadores de grupo únicos e identificadores pessoais de entidades e propaga a lista completa dos nós disponíveis atualmente. Este último serviço permite a descoberta de novos nós e o monitoramento da disponibilidade atual dos membros dos grupos.

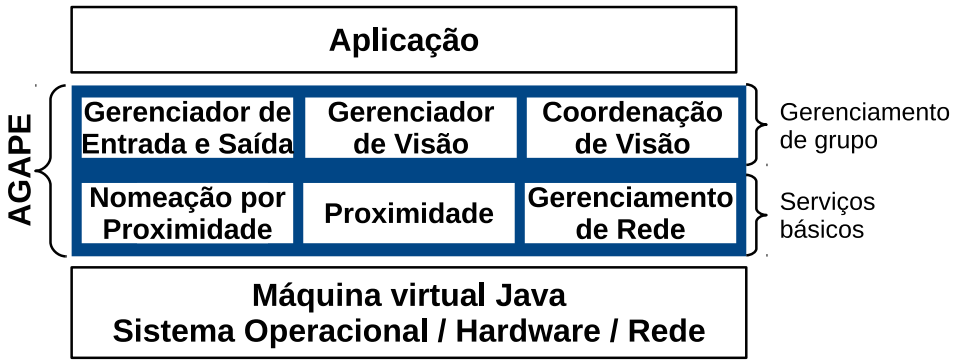



Figura 2.11: A arquitetura do AGAPE.
Fonte: Adaptado de (BOTTAZZI; CORRADI; MONTANARI, 2005)

A camada de Gerenciamento de Grupo também possui três serviços. O Gerenciador de Entrada e Saída permite que os nós descubram, participem e deixem grupos de interesse. Ele também permite que os nós promovam dinamicamente a formação de novos grupos. O Gerenciador de Visão permite que os nós criem e disseminem visões de grupos ou visões dependentes do contexto em tempos regulares. Quando os membros dos grupos entram ou saem da rede ou quando eles alteram o dispositivo de acesso e/ou perfil do grupo, o AGAPE reporta as alterações da visão para todos os membros do grupo na localidade explorando o suporte de comunicação do Serviço de Gerenciamento de Rede. Finalmente, o Serviço de Coordenação de Visões permite que os nós decidam se distribuem ou não visões dependentes de contexto. Ele auxilia a reduzir propagações desnecessárias, quando múltiplos nós pertencem ao mesmo grupo e definem a mesma localidade para disseminar a mesma visão para membros do grupo.

### 2.3.5 Transhumance (2007)

O Transhumance (DEMEURE et al., 2008) foca em redes pequenas com até 20 nós movimentando-se em velocidades de até 5 km/h. Para participar da rede, os nós devem definir previamente os seus perfis e preferências. No Transhumance, os nós podem ver grupos existentes e suas propriedades, participar de um ou mais grupos e criar novos grupos. Os membros de um grupo podem se comunicar entre si e utilizar serviços associados ao grupo, como compartilhamento de dados e bate-papo multi-usuário.

A figura 2.12 apresenta a arquitetura do Transhumance, que pode ser dividida em

cinco componentes: gerenciamento de energia, comunicações, grupos, serviços comuns e segurança. O gerenciamento de energia possui os módulos de monitoramento, políticas e decisão, que toma decisões sobre ações de adaptação a serem executadas, baseado nas política e informações coletadas.

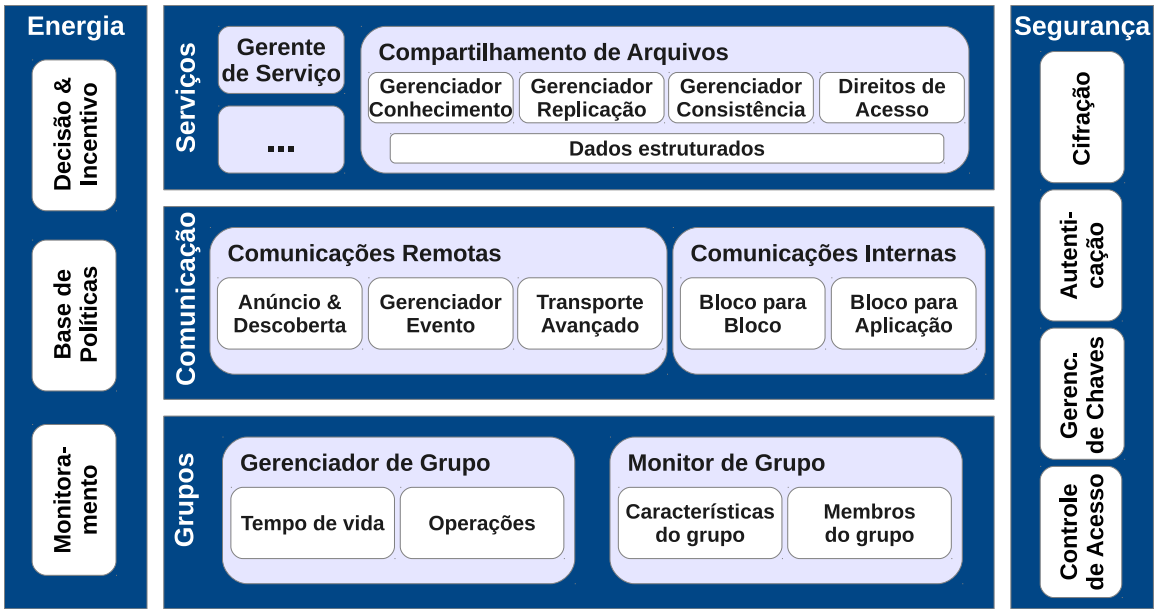



Figura 2.12: Arquitetura do Transhumance.
Fonte: Adaptado de (DEMEURE et al., 2008)

O componente de comunicação utiliza os serviços de um protocolo de roteamento reativo. O módulo de transporte é um protocolo de transporte baseado no *User Datagram Protocol* (UDP) que suporta fragmentação, reconhecimentos e cifração de mensagens. O Transhumance suporta o sistema baseado em eventos publicar/assinar que garante a persistência das mensagens. O gerenciamento de grupo é responsável pelo controle das operações e associações das comunidades de interesse.

O componente de gerenciamento de identidade, presença e *hardware* inclui quatro módulos: gerenciamento de usuário, gerenciamento de terminal, identificador e presença. O gerenciamento de usuário é responsável pela manipulação dos perfis e preferências dos usuários. O gerenciamento de terminal atua como um adaptador, abstraindo algumas funcionalidades do sistema operacional e do hardware, como as chamadas ao sistema de arquivos. O módulo identificador gerencia o identificador de *hardware* do terminal, o endereço do *Internet Protocol* (IP) e o identificador do usuário. O módulo de presença

indica quais nós estão presentes na rede e sua distância.

O componente de serviços comuns reagrupa serviços de alto nível como bate-papo, transferência de arquivos e votação, fornecendo um mecanismo de anúncio/descoberta para informar os serviços disponíveis. Por fim, o componente de segurança gerencia a segurança dos recursos dos nós, grupos e as comunicações. Ele é composto por três módulos: cifração, autenticação e gerenciamento de chaves. O módulo de cifração oferece um conjunto de funções de segurança para cifrar, decifrar e assinar dados de aplicativos, garantindo confidencialidade e integridade. O módulo de autenticação é responsável pela admissão de novos membros em um grupo. O módulo de gerenciamento de chaves distribui as chaves entre os membros de um grupo de nós que possuem dados compartilhados.

Embora este *middleware* considere diversos aspectos da segurança, ele é voltado apenas para o compartilhamento de dados em redes pequenas, de até 20 nós.

### 2.3.6 Context-aware (2007)

Em (FREY; ROMAN, 2007) é apresentado um *middleware* que integra o paradigma publicar/assinar com os requisitos das aplicações móveis cientes do contexto. Esse *middleware* estende a API publicar/assinar e enriquece os eventos e as assinaturas com informações de contexto associadas com os produtores e assinantes. Os produtores podem restringir a difusão dos eventos especificando a relevância e/ou visibilidade a um contexto. Além disso, eles podem explorar a dimensão de tempo e definir eventos persistentes que permanecem disponíveis por um tempo específico após a sua publicação. Da mesma forma, os assinantes podem assinar eventos que são relevantes em domínios de contexto especificados e originados por produtores pertencentes a um contexto particular.

O *middleware* introduz a noção de especificação de contexto que permite que um nó identifique outros nós que fazem parte do mesmo contexto, considerando as suas propriedades individuais ou de grupo. Uma propriedade individual, por exemplo, pode ser velocidade de movimentação de um nó, enquanto a propriedade de grupo pode identificar os nós que estão se movendo mais rápido do que os outros em uma região. O conjunto de nós que fazem parte de algum contexto pode variar dependendo das alterações em suas

localizações, atributos e dos outros nós na configuração do sistema.

### 2.3.7 QoS-aware Adaptive Middleware (2010)

(GHOSH et al., 2010) propõe um *middleware* para "proteger" as aplicações distribuídas das condições das redes complexas enquanto suporta os requisitos de QoS das aplicações. O *QoS-aware Adaptive Middleware* (QAM) considera o contexto das aplicações, e.g. uma aplicação de alta prioridade deveria ter acesso preferencial aos recursos da rede quando competindo com uma aplicação de baixa prioridade. O principal objetivo do QAM é fornecer um *middleware* ciente de prioridade e adaptativo que atua como um intermediário entre uma aplicação e os protocolos de rede.

O QAM fornece uma API que:

a. implementa a entrega de dados a serem utilizados pelas aplicações com prioridade baseada nos requisitos de comunicação;

b. determina o conjunto de adaptações necessárias para reagir à variações das características da rede; e

c. observa as características da rede antes de aplicar qualquer adaptação.

A figura 2.13 ilustra a arquitetura do QAM. Os seguinte componentes são presentes:

a. *Socket QAM*: implementa a API que permite que aplicações Java especifiquem os requisitos de fluxo de tráfego;

b. *Motor de adaptação*: implementa o analista da rede e fornece capacidade de adaptação;

c. *Interface de Controle (IC)*: implementa um ponto de incentivo para o Motor de Adaptação. Ele também determina a taxa de transmissão para o Socket QAM; e

d. *Monitor QoS*: implementa um observador de características da rede.

O QAM foi desenvolvido focando as aplicações de redes táticas. Esse tipo de rede suporta a comunicação em ambientes de forças militares, em que as ações são estruturadas

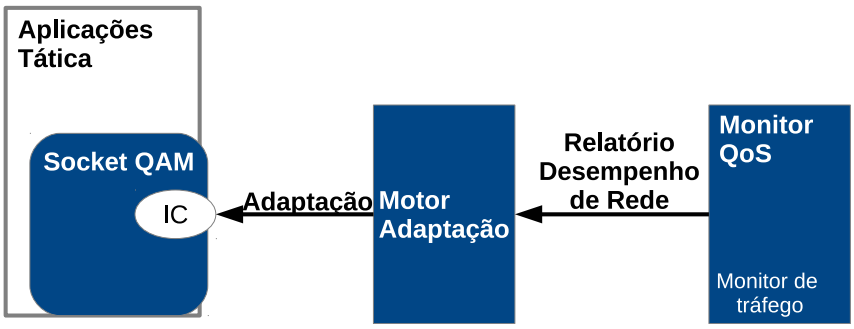



Figura 2.13: A arquitetura do QAM.
Fonte: Adaptado de (GHOSH et al., 2010)

em pelotões ou grupos. A restrição de largura de banda das redes táticas requer que as comunicações sejam priorizadas. O QAM fornece essa priorização sem impor restrições aos desenvolvedores.

### 2.3.8 Comparativo dos middleware baseados contexto

A tabela 2.4 resume as soluções de *middleware* baseadas contexto. Todas estas soluções apresentam suporte a grupo, sendo que a formação dos grupos está sempre associada aos contextos. Também, tais soluções apresentam algum método para a descoberta de recursos.

Tabela 2.4: Comparativo dos *middleware* baseados em contexto

| **Middleware** | **Suporte a grupos** | **Descoberta de recursos** | **Localização** | **Segurança** |
|---|---|---|---|---|
| STEAM | Sim, por proximidade | Sim | Sim | ND |
| SELMA | Sim, baseado em uma localidade geográfica | Sim | Sim | ND |
| EMMA | ND | ND | ND | ND |
| AGAPE | Sim, entre nós vizinhos | Sim | Sim | ND |
| Transhumance | Sim, baseado nos contextos definidos | Sim | Sim | Parcial, limitado a 20 nós |
| Context-aware | Sim, baseado nos contextos definidos | Sim | ND | ND |
| QAM | ND | ND | ND | ND |

Dentre as soluções baseadas em contexto, o EMMA, o QAM e o Context-aware (FREY; ROMAN, 2007) não apresentam técnicas para a localização de nós e recursos. Contudo, apenas o Transhumance (DEMEURE et al., 2008) relata técnicas para fornecer segurança

à comunicação. Contudo, as técnicas apresentadas pelo Transhumance são limitadas a redes pequenas, com até 20 nós.

## 2.4 Cross-layer

Algumas soluções de *middleware* consideram o uso de abordagens *cross-layer*, que integram as funcionalidades de várias camadas da pilha de protocolos em um único local. Segundo pesquisadores, essa abordagem diminui o atraso na tomada de decisões e facilita as operações em redes dinâmicas como as MANETs (LOPEZ et al., 2009).

### 2.4.1 Q (2005)

O Q (AVVENUTI; VECCHIO; TURI, 2005) é um *middleware* do tipo publicar/assinar, em que os eventos são instâncias de aplicação definidas. Tantos os produtores como os assinantes tem que especificar o tipo de eventos que eles produzem ou estão interessados. Duas características do Q são: ele permite a reconfiguração por meio de interações cross-layer; e possui filtros baseados em conteúdo usando códigos móveis. A arquitetura do Q é ilustrada na figura 2.14.

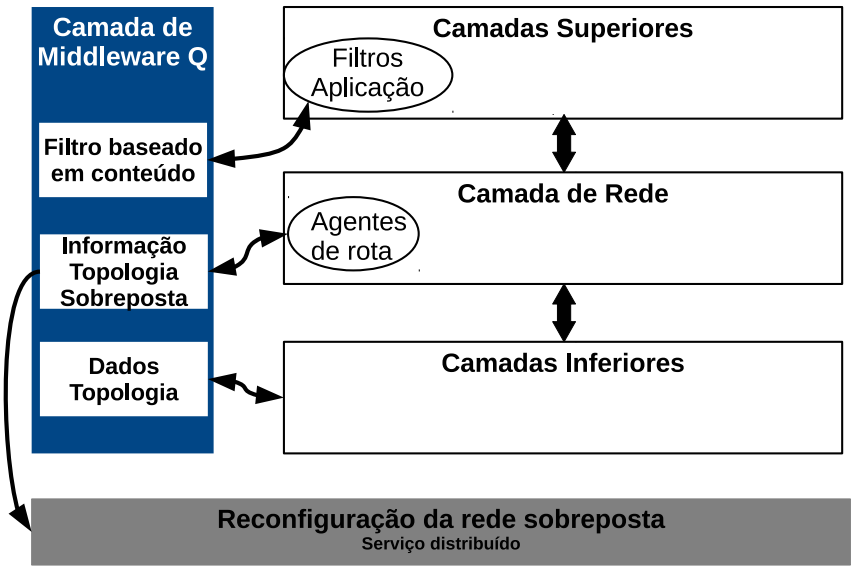



Figura 2.14: A arquitetura do Q.
Fonte: Adaptado de (AVVENUTI; VECCHIO; TURI, 2005)

O Q interage com os agentes de roteamento e usa informações da topologia para

obter uma rede sobreposta auto-reconfigurável com o objetivo de aumentar a eficiência da comunicação. Dessa forma, as rotas conectando produtores e assinantes refletem rotas *unicast* da camada de rede. No Q, os nós produtores propagam avisos de eventos e os nós interessados respondem com assinaturas. Cada aviso contém a identidade do produtor e os tipos de eventos que ele gera e deve ser periodicamente retransmitido com o objetivo de tolerar perdas de mensagens

O Q permite o filtro de eventos baseado em conteúdo: cada assinante pode especificar um filtro de conteúdo expressado com um conjunto de condições. Todos os eventos que não satisfazem a essas condições não são entregues ao assinante. O *middleware* também permite filtros derivados da composição de filtros primários.

### 2.4.2 Cooperative Caching (COCA) (2007)

O *COoperative CAching* (COCA) (TIAN; DENKO, 2007) é baseado em *clusters* e fornece um serviço de cache para aplicações de usuário. A arquitetura do COCA é ilustrada na figura 2.15 e consiste de cinco módulos básicos: *clustering*, perfil da pilha, *prefetching*, busca de informações e gerenciamento de cache.

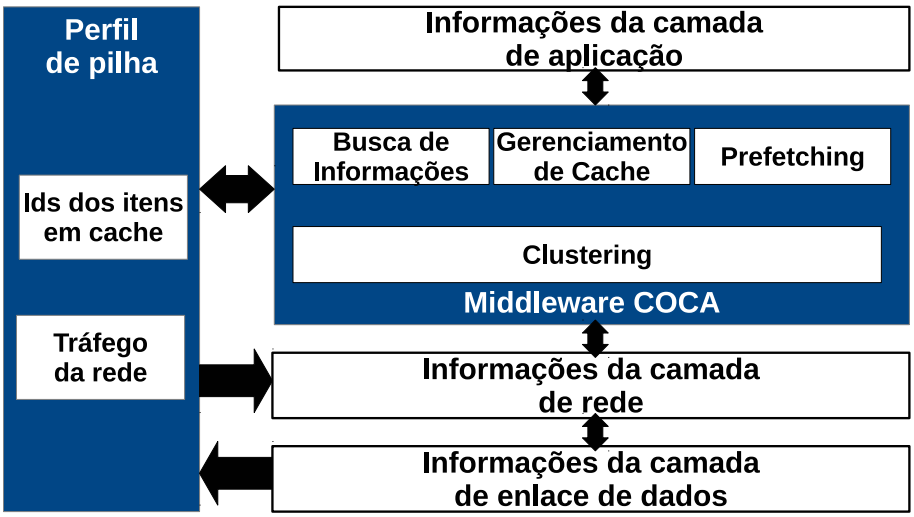



Figura 2.15: Arquitetura do COCA.
Fonte: Adaptado de (TIAN; DENKO, 2007)

O módulo *clustering* é responsável pela formação e manutenção dos *clusters*. O módulo de perfil da pilha fornece informações *cross-layer* compartilhadas entre o *middleware*, a

camada de rede e a camada de enlace de dados. O módulo *prefetching* determina quais itens de dados deveriam ser pré-carregados para serem usados em operações futuras. O módulo de busca de informações trata a localização dos itens de dados solicitados pelo cliente. Para reduzir o atraso das consultas, um nó consulta os itens solicitados dentro da vizinhança antes de enviar um pedido à origem dos dados.

O módulo de gerenciamento de cache consiste de três sub-módulos: controle de admissão, substituição e consistência. O controle de admissão determina se um dado recebido deve estar em cache ou não. A substituição determina quais itens devem ser removidos quando o cache está cheio e um novo item deve ser alocado. A consistência mantém a sincronização dos itens em cache com o dado na fonte original.

As camadas que têm alguma informação para ser compartilhada alocam essa informação no módulo de perfil da pilha, onde ela pode ser pesquisada por outras camadas. Dois tipos de informações *cross-layer* podem ser trocadas entre camadas. O primeiro é o Estado de Tráfego da Rede, fornecido pela camada de enlace e usado pelo *middleware* no processo de pré-alocação. Assim, um nó inicia uma pré-alocação apenas quando o tráfego de rede é baixo. O segundo tipo de informação são os IDs dos Itens em Cache fornecidos pelo *middleware*. Se um nó intermediário tem uma cópia de um item solicitado, ele descarta o pacote de solicitação e envia o item ao solicitante.

A principal desvantagem do COCA é que ele é apenas aplicável para o gerenciamento de cache. Além disso, como ele depende do algoritmo de *clustering*, o desempenho pode ser afetado com altas taxas de mobilidade.

### 2.4.3 MobCross (2009)

O MobCross (DENKO; SHAKSHUKI; MALIK, 2009) é baseado na publicação e assinatura de tópicos. As suas principais funcionalidades são: conhecimento da mobilidade, reconfiguração da rede, cache de mensagens e suporte a interação *cross-layer*. A arquitetura suporta comunicação P2P e a interação entre os componentes para a troca de informações entre as camadas de aplicação e de rede.

O *middleware* é composto por quatro componentes: descoberta de recursos, gerencia-

dor de dados, monitoramento de mobilidade e otimização *cross-layer*, como ilustrado na figura 2.16. Cada componente precisa de informações de outros componentes ou camadas durante a comunicação. A arquitetura também contém uma pilha *cross-layer*, que mantém informações compartilhadas entre as camadas.

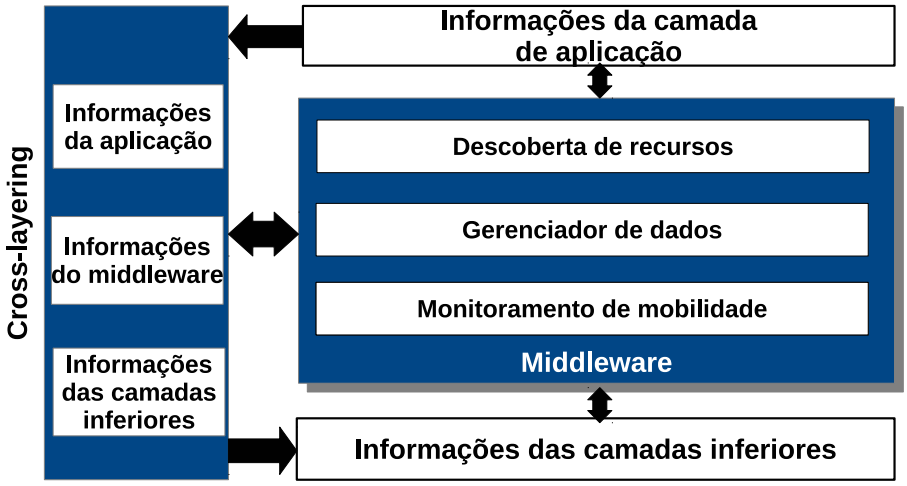



Figura 2.16: A arquitetura MobCross.
Fonte: Adaptado de (DENKO; SHAKSHUKI; MALIK, 2009)

O componente de monitoramento de mobilidade fornece informações relacionadas à mobilidade do nó e a topologia da rede. Para esse fim, o MobCross obtém informações de localização sobre os nós próximos e usa um esquema baseado em cadeias de Markov (DENKO, 2004) para a previsão de localidade.

O MobCross introduz um novo componente para troca de informações *cross-layer* (CONTI et al., 2004), uma camada vertical independente da pilha de protocolos tradicional e compartilhada por todas as camadas. As informações compartilhadas incluem localização dos nós, tabelas de roteamento, estado do enlace e recursos disponíveis. O MobCross obtém informações da camada de aplicação e combina com informações da topologia obtidas da camada de rede, para melhorar o desempenho da comunicação. A descoberta de recursos e os serviços de monitoramento da mobilidade são usados com informações de roteamento para o gerenciamento da topologia da rede e auto-reconfiguração dos nós.

O componente de descoberta de recursos é usado para descobrir recursos, redes, usuá-

rios ou dispositivos durante a desconexão da rede ou serviços. Ele pode ser usado para descobrir um nó alternativo que gerencie tópicos de interesse quando um assinante perde uma conexão inicial. Esse módulo troca informações com o módulo de monitoramento de mobilidade usando o módulo de otimização *cross-layer*. Os recursos descobertos podem ser compartilhados entre os nós.

O MobCross também fornece um serviço de cache como um mecanismo para evitar a perda de mensagens durante as falhas de enlace ou mobilidade dos nós, suportando operações desconectadas. O componente gerenciador de dados coordena o armazenamento de mensagens quando os nós se movimentam e as mensagens são devolvidas. Esse componente pode manter informações sobre os tópicos e o número de assinantes. Dessa forma, se não existe um nó assinante para a mensagem ou nenhuma nova assinatura é realizada durante um período de tempo, a mensagem será excluída do cache.

### 2.4.4 MChannel (2009)

Em (LOPEZ et al., 2009) é proposto um *middleware* ciente da topologia para suportar a comunicação em grupo nas MANETs, chamado MChannel. Neste *middleware*, o protocolo de roteamento é movido para a camada de aplicação. Com isso, segundo os autores, se obtém uma maior integração entre o *middleware* e a camada de roteamento, flexibilidade nas alterações do ambiente e simplicidade no desenvolvimento.

O MChanel modifica o JGroups do Java (BAN, 2014), uma ferramenta de comunicação *multicast* confiável que fornece associação de grupos, métodos para envio de mensagens para um ou todos os membros de um grupo, detecção e remoção de membros falhos e ouvintes de eventos. Além disso, ele utiliza serviços de *unicast* e *multicast* fornecidos pelos protocolos OMOLSR e jOLSR, também definidos em (LOPEZ et al., 2009). O OMOLSR é um protocolo *multicast* construído sobre o protocolo *unicast* OLSR (CLAUSEN; JACQUET, 2003), e o jOLSR é uma implementação Java do OLSR. A figura 2.17 apresenta a arquitetura do MChannel.

O protocolo OMOLSR calcula o roteamento *multicast* baseado em eventos recebidos do protocolo jOLSR. O jOLSR tem algumas modificações adicionadas à especificação

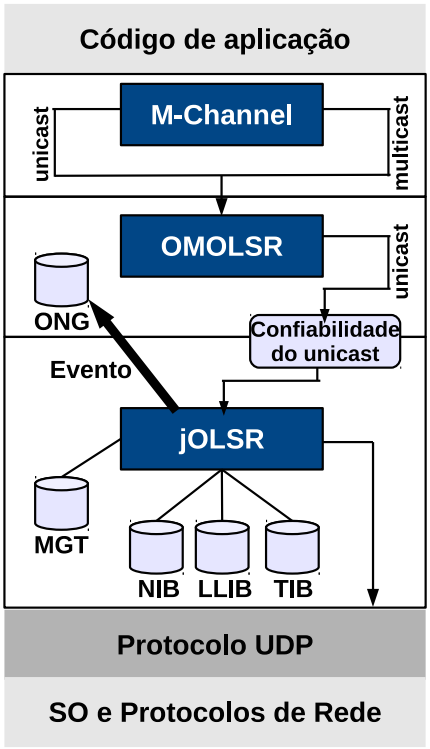



Figura 2.17: A arquitetura do MChannel.
Fonte: Adaptado de (LOPEZ et al., 2009)

básica do OLSR para fornecer informações sobre a topologia e a associação de grupo para os serviços superiores. Ele armazena informações de rede em três tabelas: Base de Informações de Vizinhos (NIB) que mantém informações sobre os vizinhos, Base de Informações da Topologia (TIB) que contém informações da topologia da rede e Base de Informações do Enlace Local (LLIB) que mantém informações atualizadas sobre os estado dos enlaces dos vizinhos.

Em cada grupo, o MChannel fornece mecanismos para suporte de associação em grupo, detecção de falhas e controle de fluxo. O serviço de **associação de grupo** é baseado em informações da árvore *multicast* OMOLSR. A detecção de falhas depende dos serviços fornecidos pelo jOLSR, que verifica a disponibilidade dos nós e repara o grafo de topologia da rede. Por fim, o controle de fluxo também é alcançado pelos serviços fornecidos pelo jOLSR.

Em (COSTAGLIOLA et al., 2012), o MChannel foi melhorado a fim de torná-lo ciente de energia e atraso. Os autores adicionaram um novo módulo ao MChannel permitindo roteamento *unicast* baseado no atraso fim-a-fim e no tempo de vida da rede. Usando essas

informações, os autores mostram que o novo MChannel aumenta o tempo de vida da rede e diminui o atraso fim-a-fim.

### 2.4.5 Comparativo dos middleware cross-layer

A tabela 2.5 resume as soluções de *middleware cross-layer*. Dentre essas soluções, apenas o MChannel apresenta o suporte a grupos, usando a ferramenta jGroups.

Tabela 2.5: Comparativo dos *middleware cross-layer*

| **Middleware** | **Suporte a grupos** | **Descoberta de recursos** | **Localização** | **Segurança** |
|---|---|---|---|---|
| Q | ND | ND | Parcial, apenas localização dos nós | ND |
| COCA | ND | ND | Parcial, apenas dentros dos *clusters* | ND |
| MobCross | ND | Sim | Sim | ND |
| MChannel | usando o jGroups | Sim | ND | ND |

O serviço de localização não é totalmente considerado apenas pelo MobCross, enquanto o Q e o COCA possuem apenas um suporte parcial. Por fim, apenas o MobCross e o MChannel discutem técnicas para a descoberta de recursos, e nenhuma das soluções discute técnicas de segurança para *middleware*. A abordagem *cross-layer* é a mais limitada encontrada na literatura. Contudo, essa abordagem poderia oferecer soluções mais completas, já que não estão restritas a uma única camada de comunicação ou protocolo.

## 2.5 Orientados a aplicação

As soluções de *middleware* orientadas a aplicação são propostas específicas para uma função. Eles foram desenvolvidos para solucionar ou ajudar na solução de uma tarefa específica, por exemplo, replicação, colaboração, segurança ou, até mesmo, compartilhamento de fotos.

### 2.5.1 REDMAN (2005)

Em (BELLAVISTA; CORRADI; MAGISTRETTI, 2005), o *middleware REplication in Dense MANETs* (REDMAN) é apresentado. Ele é um *middleware* leve para gerenciar,

recuperar e disseminar, de forma transparente para os usuários, as réplicas de dados e serviços. Os autores afirmam que o REDMAN facilita o desenvolvimento de aplicações distribuídas escaláveis para MANETs densas.

A figura 2.18 ilustra a arquitetura de duas camadas do REDMAN: a camada inferior, Configuração de MANET Densa, é composta pela Identificação e Gerenciamento de MANET Densa e pelo Gerente de Eleição, enquanto a camada superior inclui a Disseminação de Réplica, Mantenedor de Grau de Réplica e Recuperação de Réplica.

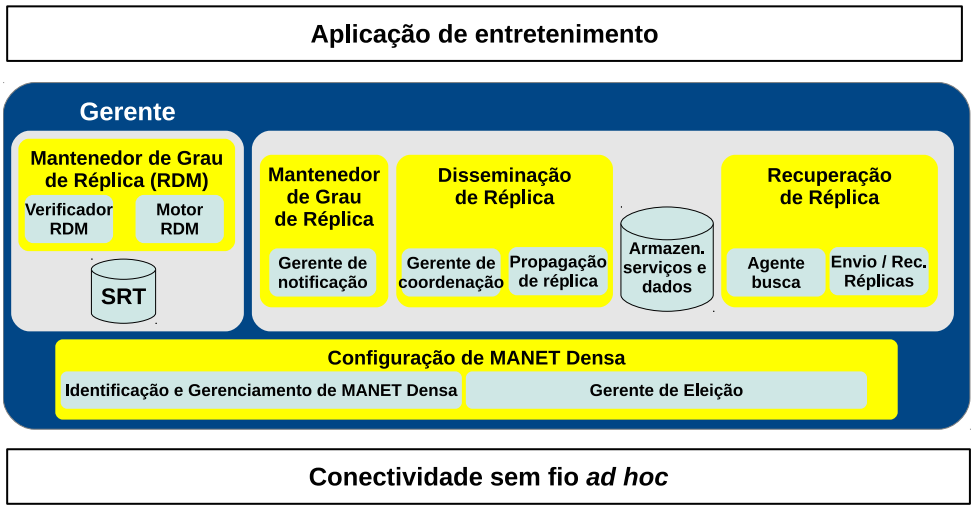



Figura 2.18: A arquitetura do REDMAN.
Fonte: Adaptado de (BELLAVISTA; CORRADI; MAGISTRETTI, 2005)

A Configuração de MANET Densa é responsável por identificar os nós e eleger gerentes de réplicas. Para essas propostas, o REDMAN emprega dois protocolos leves, projetados para MANETs densas, que impõem uma sobrecarga limitada e alcançam resultados não-ótimos mas suficientemente exatos para os serviços propostos. O detalhamento desses dois protocolos pode ser encontrado em (BELLAVISTA; CORRADI; MAGISTRETTI, 2005).

A Disseminação de Réplica distribui, de forma transparente, as réplicas na MANET. O REDMAN associa cada recurso de interesse comum com uma descrição baseada em metadados, que inclui o grau de replicação esperado. Quando um nó delegado entra em uma MANET densa, ele comunica o metadado dos recursos compartilhados para o gerente de réplicas. O gerente mantém uma Tabela de Recursos Compartilhados (SRT - *Shared Resources Tables*) com uma entrada para cada recurso gerenciado: cada entrada

contém o grau de replicação a ser garantido e a informação fracamente consistente sobre a localização da réplica.

A Recuperação de Réplica tem como objetivo recuperar eficazmente os recursos replicados. Ele executa uma recuperação simples baseada na solicitação de recursos por meio de inundações limitadas. O Mantenedor de Grau de Réplica mantém o grau de replicação decidido para cada recurso compartilhado. Após a distribuição de réplica inicial, o REDMAN reage apenas quando as réplicas deixam a MANET densa.

Uma outra solução baseada no REDMAN é apresentada (KUMAR et al., 2010), na qual os autores propõem um *middleware* de estratégia de replicação para distribuir réplicas do jogo Civilization®. Contudo, essa solução apenas altera a estratégia de replicação.

### 2.5.2 SCOMET (2007) / AGORA (2008)

Em (ARRUFAT; PARÍS; LóPEZ, 2008), (SáNCHEZ-ARTIGAS et al., 2008) e (ARRUFAT et al., 2007), os autores propõem duas arquiteturas com o objetivo de melhorar a colaboração nas MANETs, o AGORA e o SCOMET, respectivamente. Eles são compostos por três componentes (figura 2.19): um *framework* para simplificar o desenvolvimento de aplicações colaborativas; um *middleware* de colaboração para fornecer serviços de comunicação e grupo para as aplicações e uma camada de roteamento utilizada para a comunicação de grupo. Eles diferem um do outro no *framework* e na camada de roteamento, enquanto o *middleware* é sempre o mesmo. Como o objetivo deste capítulo é a apresentação dos *middlewares*, eles foram agrupados em uma única abordagem.

O *middleware* de colaboração fornece primitivas de gerenciamento de grupos, como informações de associação e canais de comunicação nomeados, bem como diferentes paradigmas de comunicação para os protocolos de camadas superiores. O serviço de gerenciamento de grupo é desenvolvido utilizando o *toolkit* JGroups. O serviço de comunicação é composto pelo canal de comunicação, que permite que os nós enviem mensagens para os membros de um dado grupo. Ele fornece funcionalidades *unicast* e publicar/assinar, sendo que o último é suportado pelo OMCAST. Sobre este componente está o serviço de nomeação e o serviço *pub/sub*. O serviço de nomeação implementa um subconjunto

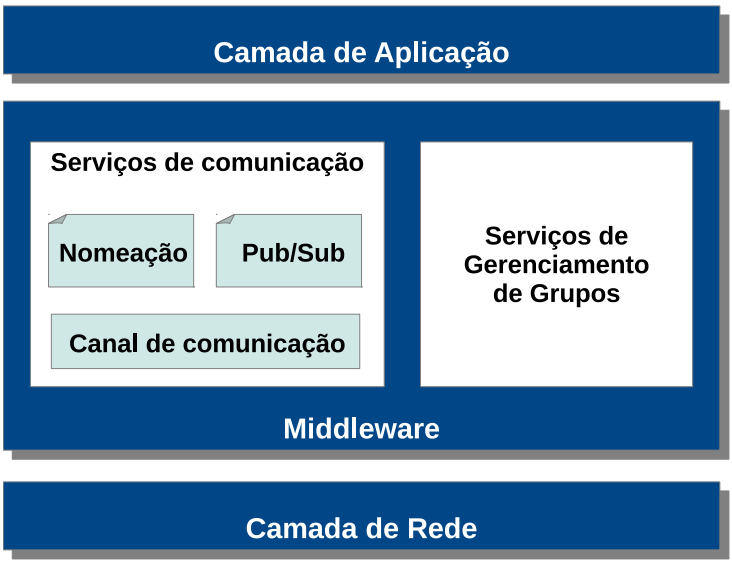



Figura 2.19: A arquitetura do SCOMET e AGORA.
Fonte: Adaptado de (ARRUFAT; PARÍS; LóPEZ, 2008; SáNCHEZ-ARTIGAS et al., 2008; ARRUFAT et al., 2007)

do *Java Naming Discovery Interfaces* (JNDI) para armazenar dados leves como descoberta de recursos e informações de grupo e coordenação. Finalmente, o serviço *pub/sub* suporta um conjunto da interface JMS para permitir que os nós publiquem e/ou assinem informações relacionadas a um dado tópico.

### 2.5.3 PASMi (2010)

(SHIFERAW et al., 2010) propõe um *middleware* chamado de *Photo Annotation and Sharing Middleware* (PASMi) para permitir que usuários nômades compartilhem e anotem fotos em MANETs. A descoberta de fotos no PASMi pode ser realizada de duas formas: *Push* e *Pull*. No *Push*, a disseminação de avisos é usada para informar os nós sobre fotos compartilhadas na área circundante. O conteúdo e a distribuição de avisos são determinados pela análise dos interesses dos usuários e suas conectividades. Já no *Pull*, os nós descobrem as fotos pesquisando em sua vizinhança. A seleção e distribuição de consultas são realizadas pelo interesse dos usuários. Quanto a anotação de fotos, o PASMi recomenda anotações por meio da análise de convivência dos usuários e do contexto das fotos.

A figura 2.20 ilustra a arquitetura do PASMi, composta de quatro estruturas de ar-

mazenamento e três módulos. O Armazenamento de Dados de Fotos contém o metadado das fotos locais. O Armazenamento de Dados da Rede contém dados históricos relacionados com o compartilhamento de fotos e atividades de anotação. A Regra Base armazena regras de associação para correlacionar o interesse dos usuários e suas conectividades com o contexto do ambiente. O Armazenamento de Dados de Avisos contém as descrições de fotos compartilhadas no ambiente.

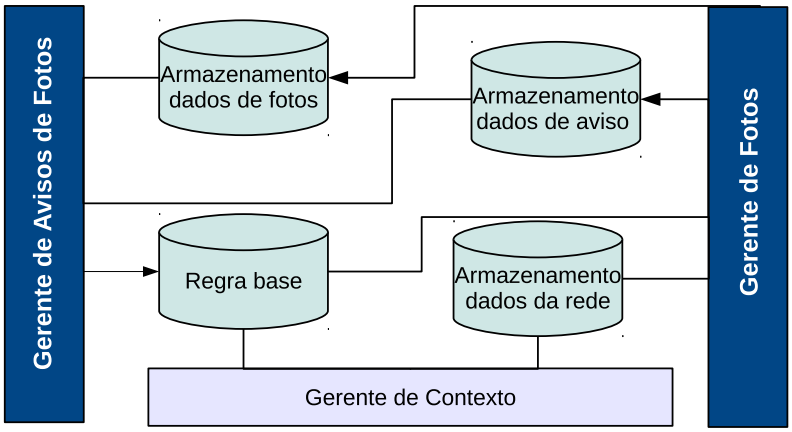



Figura 2.20: A arquiteura do PASMi.
Fonte: Adaptado de (SHIFERAW et al., 2010)

O módulo Gerente de Fotos, o núcleo do PASMi, realiza as funcionalidades do gerenciamento de fotos, como descoberta, entrega, classificação e anotação de fotos. O Gerente de Avisos de Fotos dissemina informações sobre as fotos compartilhadas aos nós na vizinhança, de acordo com sua conectividade e interesse. O Gerente de Contexto é o módulo responsável por determinar os interesses e conectividade dos usuários, por meio das regras definidas pelos usuários.

### 2.5.4 Esquemas de Chandrakant et. al (2011)

Em (CHANDRAKANT et al., 2011a; CHANDRAKANT et al., 2011b), os autores apresentam uma solução para fornecer um tipo de segurança aos serviços de *middleware*. Essas soluções não são sistemas de *middleware*, mas artefatos que poderiam ser implementados nas soluções já existentes. Os autores demonstram como restringir a admissão de nós egoístas ou maliciosos, em redes escaláveis ou não.

### 2.5.5 Esquema de Lahyani et. al (2012)

Em (LAHYANI et al., 2012), os autores apresentam uma abordagem para sistemas do tipo publicar/assinar cientes de Qualidade de Serviço. Eles têm como objetivo as aplicações de gerenciamento de crises, que visam localizar ameaças potenciais e evitá-las. Assim, tais sistemas requerem uma alta Qualidade de Serviço e precisam considerar diversos critérios que podem afetar a qualidade dos nós e os enlaces na rede. A solução proposta monitora o sistema e analisa seus estado para prever degradações da Qualidade de Serviço, e define novas ações de reconfiguração para serem aplicadas, se necessário. Todas essas informações são fornecidas pela camada do *middleware*, embora o artigo não defina um novo *middleware*.

### 2.5.6 Comparativo dos middleware orientados a aplicação

A tabela 2.6 resume as soluções de *middleware* orientadas a aplicação. Essas soluções são indicadas para fornecer um serviço específico. Contudo, elas não apresentam as características desejáveis para uma solução genérica de *middleware*, em geral. Entre as soluções, apenas o esquema de Chandrakant tem algum tipo de serviço de segurança, mas limitado a evitar a participação de nós maliciosos na comunicação. Por outro lado, apenas o SCOMET e o AGORA apresentam primitivas para permitir que o *middleware* crie e gerencie grupos de usuários.

Outros serviços orientados a aplicação podem ser encontrados na literatura, embora não sejam mencionados aqui por não serem considerados soluções de *middleware*.

## 2.6 Conclusão

Este capítulo apresentou as soluções de *middleware* desenvolvidas para as MANETs, que foram classificadas em baseadas em espaço de tuplas, baseadas em P2P, baseadas em contexto *cross-layer*, e orientadas à aplicação. Em cada abordagem foi realizado um comparativo das características das soluções apresentadas, destacando os principais serviços disponibilizados para o suporte das operações nas MANETs. Dentre as abordagens

Tabela 2.6: Comparativo dos *middleware* orientados a aplicação

| **Middleware** | **Suporte a grupos** | **Descoberta de recursos** | **Localização** | **Segurança** |
|---|---|---|---|---|
| REDMAN | ND | Sim, usando inundação limitada | Parcial, apenas na vizinhança | ND |
| SCOMET | Usando JGroups | Usando JNDI | ND | ND |
| AGORA | Usando JGroups | Usando JNDI | ND | ND |
| PASMi | ND | Por avisos e consultas distribuídas | ND | ND |
| Chandrakant et. al | ND | ND | ND | Parcial, apenas evitar a participação de nós maliciosos na comunicação |
| Lahyani et. al | ND | ND | ND | ND |

discutidas, as soluções baseadas em contexto são as que apresentam a maior variedade de serviços para as MANETs.

Diversas soluções também consideram o uso da abordagem em grupos. Essa abordagem facilita a integração da segurança (DEMEURE et al., 2008) e se adapta bem à maioria das aplicações das MANETs (KIM; MAZZOCCHI; TSUDIK, 2003). Dentre os tipos de *middleware* apresentados, todos os baseados em contexto apresentam alguma forma de suporte a grupos, considerando a vizinhança dos nós, sua localização geográfica ou o contexto das aplicações.

As soluções apresentadas neste capítulo não consideram totalmente as questões de segurança, sendo susceptíveis a ataques maliciosos. Dessa forma, elas deixam as aplicações vulneráveis a ataques comuns nesta camada, tais como Sybil, personificação e negação de serviço. Dentre as soluções de *middleware* propostas, a MESH*Mdl* (HERRMANN; MüHL; JAEGER, 2007) considera superficialmente a segurança, apresentando um mecanismo para garantir o anonimato na comunicação. No entanto, outros aspectos e serviços importantes da segurança não foram considerados. Outro *middleware* que garante a segurança no fornecimento de serviços é o Transhumance. Contudo, ele é voltado apenas para o compartilhamento de dados em redes pequenas com, no máximo, 20 nós.

No próximo capítulo é apresentada uma proposta de *middleware* seguro para MANETs baseada em contexto e que utiliza uma abordagem de grupos como suporte às suas operações. A abordagem baseada em contexto foi a utilizada porque facilita o fornecimento de

serviço às aplicações e a organização dos grupos de acordo com os seus interesses comuns ou contexto dos serviços disponibilizados. São discutidos os principais serviços fornecidos pelo *middleware* e as formas de garantir a segurança em suas operações.

# CAPÍTULO 3

# MIDDLEWARE SEGURO PARA REDES AD HOC MÓVEIS

Este capítulo descreve a proposta de um *middleware* para MANETs com o objetivo de garantir segurança aos serviços fornecidos nessas redes. Embora diversas soluções de *middleware* para as MANETs possam ser encontradas na literatura (ver capítulo 2), nenhuma delas considera, do nosso ponto de vista, totalmente os requisitos de segurança dessas redes. A figura 3.1 ilustra como as aplicações podem utilizar um *middleware* para realizarem uma comunicação confiável sobre um meio físico não-confiável.

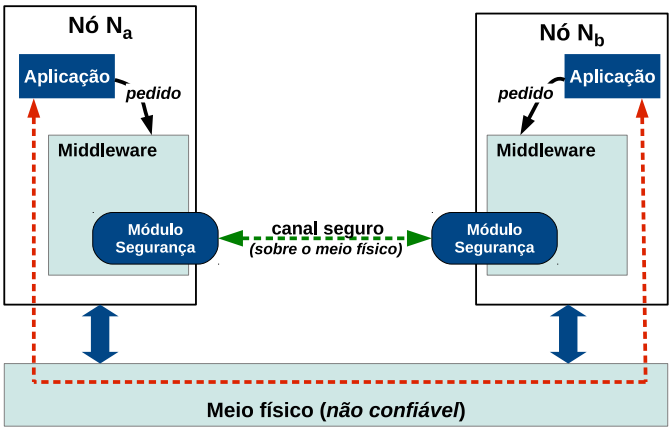



Figura 3.1: Comunicação confiável usando um *middleware* seguro

Assim, é apresentado o SEMAN, um *middleware* baseado em contexto e que utiliza uma abordagem em grupos para dar suporte às tomadas de decisão quanto à segurança. Considerando as soluções apresentadas no capítulo 2, nota-se que aquelas baseadas em contexto apresentam mais facilidade em fornecer diferentes tipos de serviços às MANETs. Além disso, acredita-se que o uso de grupos facilita a organização dos nós dentro dos contextos e as tomadas de decisão quanto à segurança. Inicialmente é apresentada uma breve descrição do *middleware* e, em seguida, todos os seus módulos e componentes são detalhados.

## 3.1 Visão Geral

Segundo (AL-JAROODI et al., 2010), um *middleware* seguro para MANETs deve considerar alguns pontos importantes, como:

a. mecanismos de autenticação e gerenciamento de credenciais;

b. gerenciamento de autorização e controle de acesso;

c. integridade e segurança dos dados compartilhados;

d. comunicação segura em grupo e par-a-par; e

e. suporte aos requisitos de ambientes heterogêneos.

Para isso, o SEMAN fornece suporte para comunicações entre múltiplos pares de forma segura e confiável em ambientes susceptíveis a ataques maliciosos. Ele fica localizado entre as camadas de aplicação e transporte, fornecendo serviços seguros às aplicações. O SEMAN é composto por módulos distribuídos e por um conjunto de operações criptográficas baseadas em grupos. Para garantir a segurança aos serviços fornecidos, são propostas medidas integradas de tolerância e prevenção a ataques maliciosos.

Para auxiliar as operações do *middleware* é utilizado um esquema de grupos, que são formados por nós que possuem características similares. Tais grupos são chamados de grupos de contexto e são formados dinamicamente e auto-organizadamente sem a interferência de usuários, apenas considerando os perfis e os requisitos das aplicações. Os serviços são fornecidos e utilizados pelas aplicações dentro de um contexto e, portanto, são facilmente disponibilizados aos nós que pertencem aos grupos deste contexto.

O SEMAN é composto por uma interface de comunicação, um catálogo e três módulos: serviços, processamento e segurança, conforme ilustra a figura 3.2. Como ilustrado, as solicitações das aplicações podem ser direcionadas ao *middleware* ou às camadas subjacentes, como as camadas de transporte ou rede. Sem perder a generalidade, assume-se que todos os pedidos das aplicações serão direcionados ao *middleware*, usando primitivas apropriadas via comunicação entre processos baseada em trocas de mensagens. Todas as

trocas de mensagens entre o *middleware* e as aplicações são realizadas usando a Interface de Comunicação, que classifica as mensagens e entrega ao módulo apropriado ou à aplicação de destino.

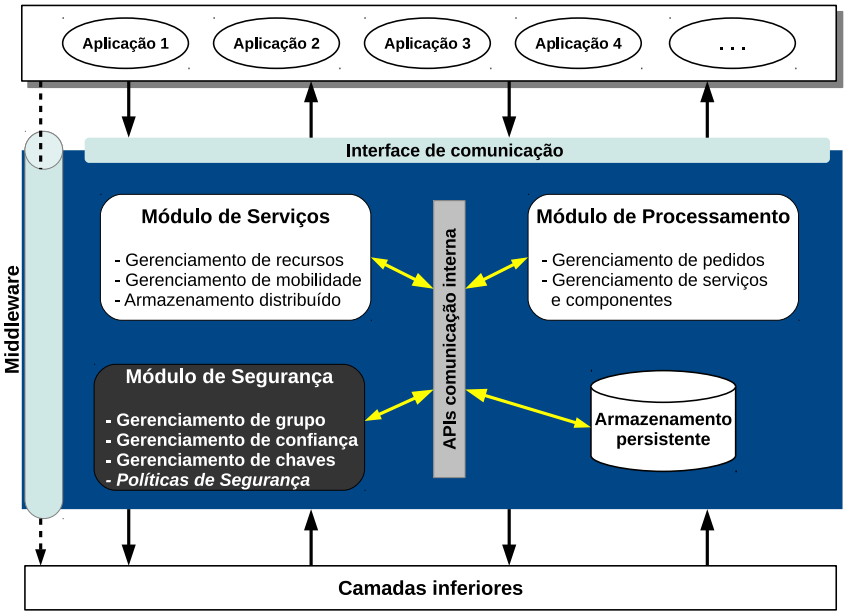



Figura 3.2: A arquitetura do *middleware* seguro

O Módulo de Serviços contém os serviços básicos que são fornecidos pelo SEMAN. Ele é formado por vários componentes que são responsáveis pelo gerenciamento de um ou mais serviços, como Gerenciamento de Recursos, Gerenciamento de Mobilidade e Armazenamento Distribuído. Essa lista de componentes não é restrita e novos componentes podem ser facilmente agregados ao SEMAN. O Módulo de Processamento é responsável por manter o funcionamento central do SEMAN. Ele é composto pelos seguintes componentes: Gerenciamento de Pedidos, Gerenciamento de Serviços e Gerenciamento de Módulos.

O Módulo de Segurança é responsável por garantir à comunicação as propriedades de segurança dentro de um limiar pré-estabelecido. Como o objetivo do SEMAN é fornecer segurança às aplicações que usufruem de seus serviços, este módulo é fundamental. Ele é composto por três componentes principais: gerenciamento de confiança, gerenciamento de chaves e gerenciamento de grupos. Todos os componentes desse módulo foram desenvolvidos pelo autor e são apresentados nos próximos capítulos.

O catálogo é composto por uma memória não-volátil e é responsável por manter todos os pedidos pendentes e informações de segurança sobre as aplicações e nós, tais como chaves criptográficas, informações de confiança, credenciais, etc. Ele é importante para garantir resiliência em, no mínimo, três situações: (*i*) falha física ou reinício do nó; (*ii*) desconexão da rede; e (*iii*) longos atrasos no fornecimento de serviços. Essas situações podem resultar de uma ação maliciosa ou podem ser resultado do comportamento dinâmico das MANETs.

As próximas sessões detalham as principais características e funcionalidades dos três núcleos e seus componentes. São discutidos os objetivos principais desses módulos e como eles são alcançados no desenvolvimento final do SEMAN.

## 3.2 Modelo de ataques

O SEMAN visa uma rede assíncrona formada por $n$ nós móveis, representados por $N_1, N_2, ..., N_n$. Nas fases de inicialização dos grupos, assume-se que apenas nós confiáveis participam dessas atividades. O *middleware* tem com objetivo proteger a rede contra alguns tipos de ataques maliciosos, sendo eles: egoísmo, bizantino, personificação e *Sybil*. Embora outros ataques possam ser encontrados nas MANETs, nesta tese esses foram os considerados.

A seguir, são descritos esses comportamentos maliciosos, que os nós podem apresentar enquanto fornecem serviços no SEMAN. São apresentadas também as principais estratégias que são utilizadas para impedir a ação maliciosa dos atacantes contra o sistema.

### 3.2.1 Ataques de Egoísmo

Um nó pode agir de forma egoísta, tanto como consequência de um ato malicioso e proposital como de forma mal intencionada, com o simples objetivo de economizar recursos próprios. Contudo, independente do motivo, o comportamento egoísta pode comprometer as atividades da rede e as tomadas de decisão que necessitam da cooperação dos nós que formam um grupo.

Para garantir a segurança contra o comportamento egoísta dos nós, todas as operações de grupo são estruturadas considerando a técnica de compartilhamento de segredo $t$ sobre $n$, sendo que $n - (t + 1)$ nós podem estar indisponíveis, ou terem um comportamento egoísta, que o sistema ainda é capaz de atender às requisições.

Além disso, o componente de gerenciamento de confiança fornece informações sobre o comportamento dos nós dentro de um dado contexto. Assim, caso um nó tenha um comportamento egoísta, negando participar das atividades de um grupo, os demais nós do sistema podem ficar cientes desse comportamento por meio do módulo de gerenciamento de confiança.

### 3.2.2 Ataques Bizantinos

Um nó malicioso pode realizar um ataque bizantino contra o sistema, emitindo informações falsas ou, ainda, tomando decisões em nome de um grupo que não atendam aos requisitos e desejos dos demais membros. Dessa forma, um ataque bizantino pode comprometer a confiabilidade das operações do *middleware*.

A estratégia de organizar os nós em grupos considerando a técnica de compartilhamento de segredo $t$-sobre-$n$ também tem como objetivo aumentar a proteção do sistema contra ataques bizantinos. Nesse caso, um nó malicioso deveria comprometer outros $t$ nós para poder realizar alguma atividade maliciosa em nome de um grupo, o que torna a sua ação mais limitada e difícil.

Além disso, como contra o ataque de egoísmo, o gerenciamento de confiança fornece meios para que os nós informem os demais membros do sistema caso eles detectem algum comportamento bizantino em um nó malicioso. Dessa forma, com base nas informações do gerenciamento de confiança, os nós bizantinos podem ser isolados dos demais nós.

### 3.2.3 Ataques de Personificação

Um atacante também pode roubar a identidade de um nó confiável. Assim, ele pode comprometer a confiabilidade do sistema pois pode emitir de informações falsas em nome de um grupo, por exemplo. Também, nos serviços que são fornecidos por meio dos *mid-*

*dleware*, esse tipo de atacante pode realizar atividades em nome de outro membro do sistema.

O componente de gerenciamento de chaves é proposto para impedir esse tipo de ataque contra o sistema. Todos os serviços seguros fornecidos pelo *middleware* fazem uso da criptografia. Por meio deste componente de gerenciamento de chaves, o *middleware* garante que uma identidade pertence, de fato, ao nó que a está utilizando. Assim, um nó atacante precisaria comprometer todo o módulo de gerenciamento de chaves para ter sucesso na sua ação maliciosa.

Também o serviço de comunicação segurança do componente de gerenciamento de grupo aumenta a confiança do SEMAN contra esses ataques de personificação, garantindo que somente os membros de um grupo fechado serão capazes de decifrar uma mensagem transmitida a este grupo.

### 3.2.4 Ataques Sybil

Em um ataque *Sybil*, um nó malicioso cria uma identidade falsa e consegue a autorização dos demais nós para que esta identidade seja aceita no sistema. Com isso, a confiabilidade do sistema é afetada, já que um único nó pode realizar várias atividades em nome do grupo, inclusive alterando o comportamento das tomadas de decisão desse grupo.

Da mesma forma que no ataque de personificação, o componente de gerenciamento de chaves ajuda a impedir a ação de um atacante *Sybil*. Como a identidade de um nó é validade pelo gerenciamento de chaves, é necessário o comprometimento de todo o sistema para que um nó possa criar uma identidade falsa e fornece um par de chaves pública e privada válido para a nova identidade.

Também o serviço de comunicação segurança do componente de gerenciamento de grupo aumenta a confiança do SEMAN contra esses ataques. Como o serviço de comunicação segura garante que somente os membros de um grupo fechado serão capazes de decifrar uma mensagem transmitida a este grupo, ele impede que um nó que crie uma identidade false utilize essa identidade para receber mensagens destinadas aos membros

de um grupo que ele não seja participante.

## 3.3 Módulo de Serviços

O módulo de serviços é responsável por manter todos os serviços e aplicações que são disponibilizados pelo nó hospedeiro a outros nós da rede. Ele compreende os componentes de gerenciamento de recursos, gerenciamento de mobilidade e armazenamento distribuído. Todos esses componentes são gerenciados pelo módulo de serviços e são acessados diretamente pelas aplicações internas e externas ao nó hospedeiro.

Nesta tese, os componentes do módulo de serviços não foram desenvolvidos. As próximas seções apresentam algumas características que eles devem possuir e quais os serviços desejáveis que eles deveriam oferecer às aplicação que utilizam o *middleware*.

### 3.3.1 Gerenciamento de Recursos

É muito importante para as MANETs um serviço que forneça informações sobre a localização e disponibilidade dos recursos, como nós, serviços remotos e conteúdos (CHLAMTAC; CONTI; LIU, 2003). Este serviço deve respeitar algumas limitações, tais como: (*i*) minimizar a sobrecarga de comunicação, evitando atualizações desnecessárias sobre os recursos disponíveis; (*ii*) ser independente da posição geográfica dos nós; e (*iii*) ser independente do protocolo de roteamento. Este módulo deve considerar a descoberta e a alocação de recursos, bem como o gerenciamento da localização destes recursos.

O componente de Gerenciamento de Recursos deve oferecer, no mínimo, quatro subcomponentes, como ilustrado na figura 3.4: alocação, registro, descoberta e localização de recursos. Cada um desses sub-componentes requisita e fornece informações para os componentes dos módulos de processamento e de segurança. Por exemplo, o módulo de segurança fornece informações sobre a autorização dos nós e aplicações na alocação de recursos do sistema, enquanto o sub-componente de alocação de recursos deve fornecer informações sobre a utilização de recursos para o componente de gerenciamento de pedidos.

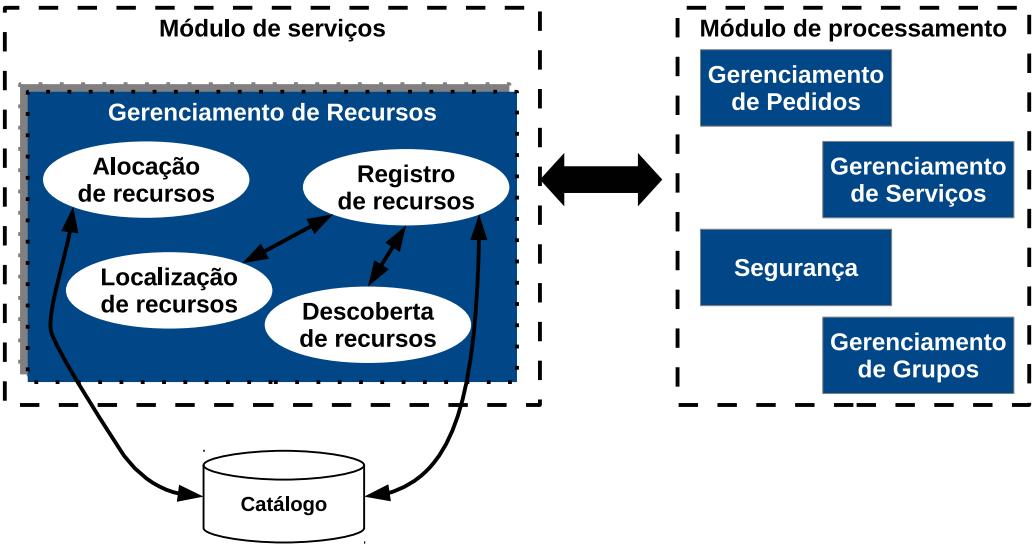



Figura 3.3: Componentes do Módulo de Gerenciamento de Recursos

As informações sobre os recursos são armazenadas localmente e são acessíveis a todas as aplicações locais que utilizam os serviços do *middleware*. Além disso, estas informações também podem ser disponibilizadas a outros nós, considerando o seu contexto e a permissão de acesso dos nós. O controle de acesso é mantido pelo módulo de segurança e é baseado na formação dos grupos de contexto.

### 3.3.2 Gerenciamento de Mobilidade

Este componente é particularmente importante, pois os nós móveis podem mudar suas posições geográficas constantemente, o que pode afetar o desempenho das aplicações distribuídas. Além da mobilidade dos nós, ele também deve considerar a mobilidade das aplicações, que podem migrar de um nó para outro durante as operações da rede. Para fornecer um serviço efetivo às aplicações, ele deve possuir três sub-componentes fundamentais: gerenciamento de localização, gerenciamento de transferência e gerenciamento de desconexão.

O gerenciamento de localização deve fornecer informações sobre a localização física dos nós às aplicações. Para isso, considera-se que os membros de um grupo mantenham suas informações de localização disponíveis neste grupo. O gerenciamento de transferência deve permitir que as aplicações móveis mantenham conexão durante a migração das aplicações

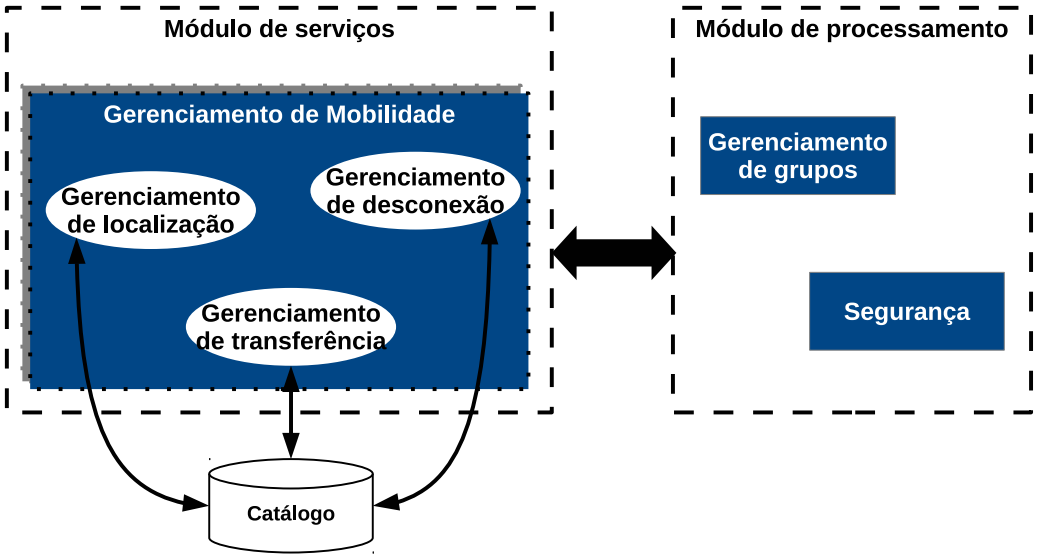



Figura 3.4: Componentes do Módulo de Gerenciamento de Mobilidade

e serviços entre nós. Ele tem como objetivo minimizar o atraso de transferência das aplicações e eliminar as perdas das informações que podem ocorrer na migração das aplicações.

Por fim, o gerenciamento de desconexão deve fornecer informações sobre a alcançabilidade ou de desconexão dos nós que fornecem serviços para o SEMAN.

### 3.3.3 Armazenamento Distribuído

Este componente deve permitir que os nós armazenem suas informações de forma distribuída, segura, dinâmica e auto-organizada na rede. Ele não depende da permanência de qualquer nó específico no sistema e deve ser altamente resistente a ataques maliciosos. Seu principal objetivo é distribuir as informações de um contexto a um grupo de nós relacionados a este contexto. Além disso, tais informações são fragmentadas pela rede, de forma que a ausência de alguns nós não afete a recuperação dos dados armazenados.

Ele é composto por quatro sub-componentes, ilustrados na figura 3.5: distribuição dos dados, recuperação dos dados, gerenciamento de réplicas e exclusão dos dados. Todos esses componentes possuem relacionamento com os módulos de segurança e gerenciamento de grupos do núcleo de processamento.

O componente de distribuição dos dados é responsável pela disseminação das informa-

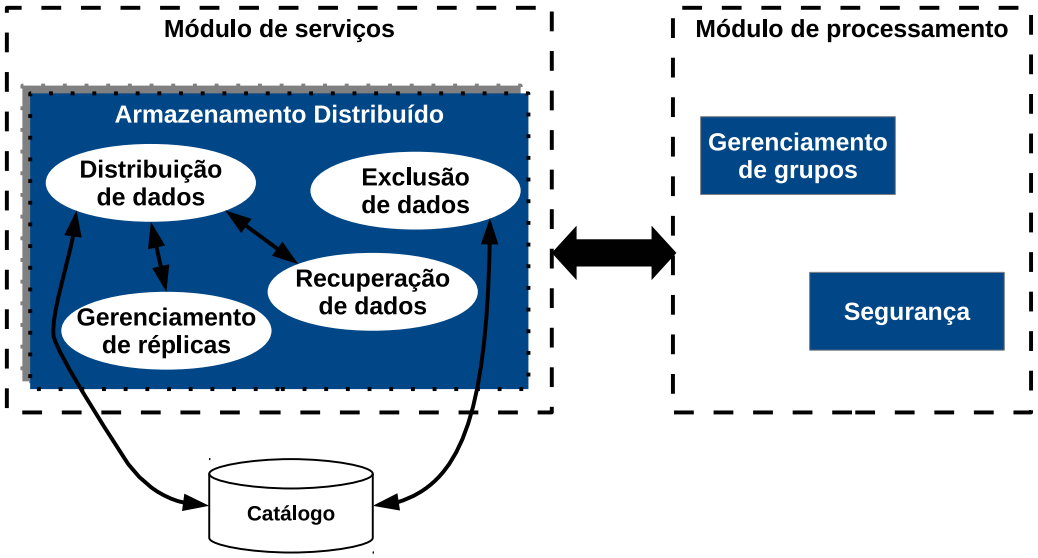



Figura 3.5: Componentes do Módulo de Armazenamento Distribuído

ções nos nós remotos. A recuperação de dados trata as requisições de acesso e localiza os dados armazenados remotamente. O gerenciamento de réplicas é responsável por manter ativas réplicas suficientes para garantir a disponibilidade das informações e por garantir a consistência desses dados. Por fim, o componente de exclusão dos dados deve garantir que, quando solicitado, um dado seja excluído de todos os hospedeiros remotos em que ele está armazenado.

## 3.4 Módulo de Processamento

O Módulo de Processamento é responsável por manter o funcionamento central do SEMAN. Ele é composto pelos componentes de Gerenciamento de Pedidos e Gerenciamento de Serviços e Componentes.

### 3.4.1 Gerenciamento de Pedidos

Este componente é responsável por manter um registro de todos os pedidos de serviços solicitados ao *middleware* pelas aplicações. Ele mantém os registros tanto dos pedidos em espera como dos já atendidos.

Uma aplicação é capaz de usar, simultaneamente, um ou mais serviços fornecidos pela rede. Devido às características altamente dinâmicas das MANETs, esta aplicação pode

não estar ciente de quais serviços estão sendo fornecidos em cada momento ou em quais nós estão hospedados esses serviços. A figura 3.6 ilustra como deve funcionar a solicitação de um serviço ao SEMAN. Ao receber uma solicitação, o componente de Gerenciamento de Pedidos obtém os parâmetros de segurança junto ao módulo de segurança. Então, ele deve verificar a disponibilidade do serviço solicitado junto ao componente de Gerenciamento de Recursos. Caso o serviço esteja sendo fornecido pelo *middleware*, ele armazena as informações sobre a solicitação no catálogo do *middleware*, faz as comunicações necessárias com os demais componentes e envia o pedido para os hospedeiros corresponentes na rede.

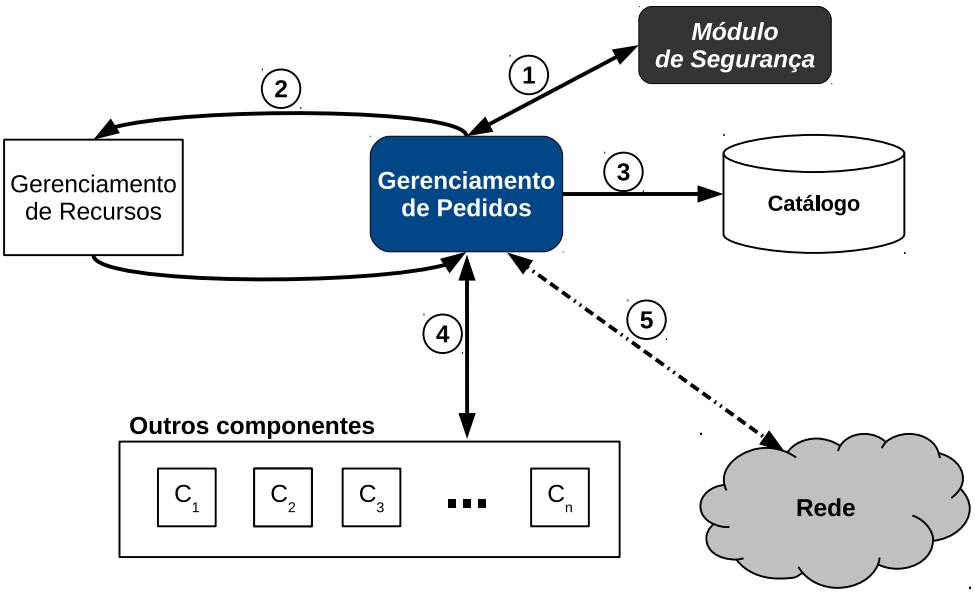



Figura 3.6: Solicitação de serviços

Como os serviços podem ser fornecidos por mais de um nó, o SEMAN pode:

a. solicitar o serviço de todos os nós que o fornecem, aumentando a disponibilidade do serviço e reduzindo o tempo de resposta;

b. distribuir os pedidos entre os nós que fornecem o serviço, balanceando a carga entre os nós;

c. escolher o nó mais confiável baseado em experiências anteriores.

Durante o fornecimento do serviço, o *middlware* pode fornecer mecanismos para prevenir ataques maliciosos. Ele deve autenticar e autorizar corretamente a aplicação. Além

disso, todas as mensagens trocadas com o *middleware* devem ser cifradas, para prevenir a escuta não-autorizada.

### 3.4.2 Gerenciamento de Serviços e Componentes

Este módulo tem uma função simples porém fundamental ao bom funcionamento do *middleware*. Ele é responsável por manter o registro de todos os serviços e componentes que estão sendo fornecidos pelo SEMAN. Quando um usuário deseja disponibilizar um serviço na rede, por meio do *middleware*, este serviço deve ser previamente registrado. Todas as informações relevantes deste novo serviço, tais como políticas de segurança e contexto de disponibilidade devem ser armazenadas no catálogo. Com isso, os demais nós da rede podem ser informados sobre a disponibilidade de um novo serviço no SEMAN.

Esse componente deve oferecer primitivas para o registro de novos serviços no *middleware*, bem como para a consulta dos serviços que estão sendo ofertados. Da mesma forma, para cada componente registrado é necessário armazenar informações sobre as formas de acesso aos serviços fornecidos por esse componente e quais o requisitos desses serviços. Por exemplo, o componente de armazenamento distribuído pode oferecer um serviço de recuperação de arquivos baseado no conteúdo do arquivo. Assim, este componente precisa gerenciar as várias formas de acesso aos serviços fornecidos pelo *middleware* com o objetivo de facilitar a integração das aplicações.

## 3.5 O Módulo de Segurança

Este módulo é o ponto principal do *middleware* e o foco desta tese. Os componentes deste módulo, ilustrados na figura 3.7, incluem: gerenciamento de chaves, gerenciamento de confiança e gerenciamento de grupos. Todos esses componentes foram desenvolvidos pelo autor e são descritos nos próximos capítulos. Eles funcionam em conjunto com os componentes de operações criptográficas e políticas de segurança, que fornecem as primitivas básicas de segurança ao módulo.

Os serviços de segurança deste módulo usam a abordagem de grupos de contexto

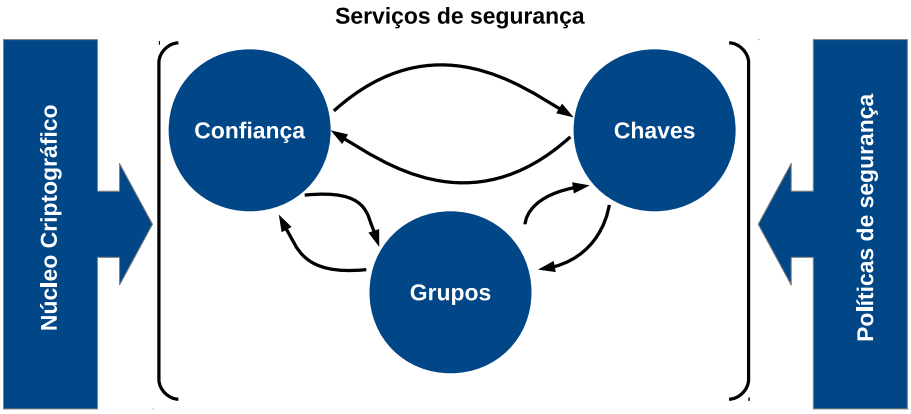



Figura 3.7: Diagrama do Módulo de Segurança

empregada nos demais serviços do *middleware*. Todas as operações de gerenciamento e tomadas de decisão neste módulo são baseadas em informações fornecidas por outros membros do grupo de contexto. Dessa forma, os nós cooperam entre si para aumentar a confiabilidade dos serviços disponibilizados na rede.

É importante ressaltar que o uso de todos os componentes do módulo de segurança não é obrigatório. A decisão de uso de um componente depende dos requisitos do usuário e das aplicações.

### 3.5.1 Núcleo criptográfico

Para garantir que as mensagens não estejam vulneráveis a ataques passivos de escuta, todas as mensagens devem ser cifradas. Dessa forma, é muito importante que a criptografia esteja presente no SEMAN. Embora qualquer mecanismo criptográfico possa ser usado, acredita-se que os mais indicados são os Criptossistemas Baseados em Identidade (*Identity-Based Cryptosystems*(IBCs)) (SILVA et al., 2008). O esquemas simétricos impõem um alto custo para gerenciar os pares de chaves secretas, sendo recomendados apenas em ambientes específicos e mais previsíveis. Além disso, se comparado com os esquemas assimétricos tradicionais, baseados em certificados, um IBC apresenta no mínimo três vantagens (CHIEN; LIN, 2008):

a. não requer certificados, eliminando o custo de armazenamento, distribuição e verificação dos certificados;

b. facilita os acordos de chaves não-interativos, reduzindo a sobrecarga de comunicação e de processamento; e

c. remove a necessidade de obter e autenticar a chave pública dos destinatários antes de enviar uma mensagem cifrada.

Outra característica importante dos IBCs é que um par de nós é capaz de computar uma chave simétrica de forma não-interativa. Esta chave pode ser usada nos esquemas de cifração autenticada.

Outra vantagem para o uso dos IBCs nas MANETs é que eles possuem um processo de gerenciamento de chave simples e um custo de armazenamento reduzido, quando comparado com outros métodos. Nos esquemas baseados em identidade, a identidade do nó ou do usuário, como um endereço de e-mail ou IP, é usada para derivar a chave pública desse nó. Assim, todo nó é capaz de descobrir a chave pública de outro nó sem a troca de nenhum dado. O Apêndice B apresenta uma visão geral do funcionamento dos criptossistemas baseados em identidade.

Por outro lado, os criptossistemas baseados em identidade apresentam uma desvantagem. A chave privada é gerada e disponibilizada, a partir da chave pública, por uma entidade conhecida como *Private Key Generator* (PKG). Esta característica impõe um desafio na implementação dos IBCs, pois o PKG pode tornar-se um ponto de falhas nas MANETs. Para mitigar o impacto de um PKG central, são propostas soluções em que o PKG é distribuído pela rede.

Diversas soluções de segurança que empregam as técnicas de IBC foram desenvolvidas para as MANETs, entre elas (CHIEN; LIN, 2008; BOHIO; MIRI, 2004; CAI et al., 2007; DENG; MUKHERJEE; AGRAWAL, 2004; HOEPER; GONG, 2006a; KHALILI; KATZ; ARBAUGH, 2003; LIU, 2006; PAN et al., 2007; PARK; LEE, 2005; SAXENA; TSUDIK; YI, 2005). Uma comparação destas soluções pode ser encontrada em (SILVA et al., 2008).

#### 3.5.1.1 Primitivas criptográficas

Grande parte dos esquemas criptográficos propostos consideram primitivas criptográficas como suporte ao seu funcionamento. Assim, essa seção apresenta as principais primitivas criptográficas necessárias para a implementação de esquemas de IBC nas MANETs.

## Problemas Computacionalmente Difíceis

Considera-se $\mathbb{G}^+$ um grupo aditivo de pontos em uma curva elíptica $E/\mathbb{F}_p$ gerada a partir de um gerador $G$ de ordem prima $q$. Também, considera-se $\mathbb{G}^\times$ um grupo cíclico multiplicativo de um corpo finito $\mathbb{F}_{p^2}$ com a mesma ordem $q$. Um mapeamento $\hat{e} : \mathbb{G}^+ \times \mathbb{G}^+ \rightarrow \mathbb{G}^\times$ é chamado de bilinear se satisfaz a igualdade $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$, para todo $P, Q \in G^+$ e $a, b \in \mathbb{Z}_q^*$.

Um mapeamento bilinear é chamado de admissível se satisfaz as seguintes propriedades:

a. **bilinearidade**: Um mapa $\hat{e} : \mathbb{G}^+ \times \mathbb{G}^+ \rightarrow \mathbb{G}^\times$ é bilinear se $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ para todo $P, Q \in \mathbb{G}^+$, e $a, b \in \mathbb{Z}_q^*$;

b. **não degeneração**: O mapa não leva todos pares de $\mathbb{G}^+ \times \mathbb{G}^+$ para a identidade de $\mathbb{G}^\times$. Como $\mathbb{G}^+$ e $\mathbb{G}^\times$ são grupos de ordem prima, se $P$ é um gerador de $\mathbb{G}^+$ então $\hat{e}(P, P)$ é um gerador de $\mathbb{G}^\times$; e

c. **computabilidade**: Existe um algoritmo eficiente, de complexidade de tempo polinomial, para computar $\hat{e}(P, Q)$ para qualquer $P, Q \in \mathbb{G}^+$.

Por fim, emparelhamento é um tipo de mapeamento bilinear admissível. Os tipos de emparelhamento mais comuns aplicados em criptografia baseada em identidade são Weil (WEIL, 1940) e Tate (TATE, 1956-1958).

Geralmente, na proposta de sistemas criptográficos, os autores selecionam alguns problemas considerados difíceis para suportar algumas afirmações quanto à segurança do sistema a ataques. No caso dos criptossistemas baseados em identidade, que baseiam-se na aplicações de técnicas de emparelhamento, são considerados os seguinte problemas computacionais de solução difícil:

a. Problema Diffie-Hellman Computacional (CDH): dado $P \in \mathbb{G}^+$, $aP$, $bP$ e $cP$ para valores desconhecidos de $a, b, c \in \mathbb{Z}_q^*$, não existe um algoritmo eficiente para computar $\hat{e}(P, P)^{abc}$;

b. Problema Diffie-Hellman Bilinear (BDH): dado $P \in \mathbb{G}^+$, $aP$, $bP$ e $cP$ para valores desconhecidos de $a, b, c \in \mathbb{Z}_q^*$, não existe um algoritmo eficiente para computar $\hat{e}(P, P)^{abc} \in G^{\times}$; e

c. Problema de Decisão Diffie-Hellman Bilinear (DBDH): dado $P \in \mathbb{G}^+$, $aP$, $bP$ e $cP$ para valores desconhecidos de $a, b, c \in \mathbb{Z}_q^*$, não existe um algoritmo eficiente para decidir se um dado $y \in \mathbb{G}^{\times}$ satisfaz $y \stackrel{?}{=} \hat{e}(P, P)^{abc}$.

## Funções Hash

Formalmente, um algoritmo *hash* é definido como um par de algoritmos de tempo polinomial probabilísticos $(Gen, H)$ que satisfaz:

a. o algoritmo de geração da chave $(Gen)$ recebe como entrada um parâmetro de segurança $(1^n)$ e gera uma chave $k$ (e.g., $k \leftarrow Gen(1^n)$); e

b. existe um polinômio $l(n)$ tal que $H_k(x)$ considerando a chave $k$ e uma *string* $x \in 0, 1^*$ gera uma *string* $H_k(x) \in 0, 1^{l(n)}$.

Uma colisão em uma função $f(x)$ é definida pela existência de dois valores $x_1$ e $x_2$ tal que $f(x_1) = f(x_2)$. Devido ao aspecto da compressão dos algoritmos *hash*, as colisões existem. Uma função é resistente a colisão se a probabilidade de encontrar intencionalmente uma colisão é desprezível. A exigência de um algoritmo *hash* seguro é fazer com que seja difícil para um algoritmo de tempo polinomial probabilístico encontrar uma colisão em um limite de tempo "razoável".

A resistência a colisões em algoritmos *hash* é uma propriedade muito forte e tipicamente difícil de ser obtida. Outras noções relaxadas de segurança incluem (DEFRAWY, 2010):

a. resistência à segunda inversão: um algoritmo *hash* é considerado resistente à segunda inversão se dadas a chave $k$ e a entrada $x_1$, é inviável para qualquer algoritmo probabilístico de tempo polinomial encontrar um valor $x_2 \neq x_1$ tal que $H_k(x_1) = H_k(x_2)$; e

b. resistência à primeira inversão: um algoritmo *hash* é considerado resistente à primeira inversão se dada uma chave $k$ e o *hash* $H_k(x_1)$ para um valor $x_1$ escolhido aleatoriamente, é inviável para qualquer algoritmo de tempo polinomial probabilístico encontrar o valor de $x_2$ tal que $H_k(x_1) = H_k(x_2)$. Nesse caso, não existe qualquer restrição de $x_2$ ser igual ou diferente de $x_1$.

Diversos algoritmos *hash* seguros foram propostos. Entre eles, o *Secure Hash Algorithm* (SHA) é o mais amplamente utilizado. Esse algoritmo foi desenvolvido pelo *National Institute of Standards and Technologies* (NIST) e é o padrão norte-americano para processamento de informações (STALLINGS, 2009). A versão atual aprovada pelo NIST é o SHA-3, originalmente denominada Keccak.

## Criptografia de limiar

Grande parte dos esquemas de IBC para as MANETs utilizam a criptografia de limiar (SHAMIR, 1979) em suas operações, especialmente o gerenciamento de chaves. A criptografia de limiar foi proposta por Shamir como uma solução para o problema de compartilhar um segredo entre um número determinado de usuários (SHAMIR, 1979). Aplicando essa técnica criptográfica, um dado $D$ pode ser dividido em $n$ partes, sendo que $D$ pode ser reconstruído apenas com $t$ partes. No entanto, com posse de $t-1$ partes completas não é possível se obter qualquer informação sobre $D$.

Assim, a criptografia de limiar $(t, n)$ resolve esse problema utilizando interpolação polinomial: considerando $t$ pontos em uma dimensão plana $(x_1, y_1), (x_2, y_2)...(x_t, y_t)$, com $x_i$'s distintos, existe um e apenas um polinômio $q(x)$ de grau $t-1$ de forma que $q(x) = y_i$. Para dividir $D$ em $n$ partes, um polinômio $q(x) = a_0 + a_1x + ... + a_tx^{t-1}$ é escolhido aleatoriamente, sendo que $a_0 = D$ e cada parte é o valor de polinômio nos $n$ pontos:

$D_1 = q(1), ..., D_i = q(i), ..., D_n = q(n)$. Assim, qualquer subconjunto de $t$ partes pode determinar os coeficientes do polinômio usando, por exemplo, interpolação de Lagrange. Com base nesses coeficientes é possível determinar o dado secreto de um certo ponto.

A criptografia de limiar também considera o uso de arimética modular, visto que conjunto de inteiros módulo um número primo $p$ forma um corpo em que a interpolação é possível. Isso tem sido amplamente empregado na construção de PKGs distribuídos para as MANETs.

#### 3.5.1.2 Operações criptográficas

Esta tese considera o uso de esquemas criptográficos baseados em identidade. Qualquer esquema baseado em identidade por ser utilizado, dependendo das necessidades do *middleware*. Sem a perda da generalidade, é empregada a técnica apresentada por Boneh e Franklim (BONEH; FRANKLIN, 2001).

Os principais algoritmos para a realização das operações criptográficas e o suporte à comunicação segura entre os nós são: configuração, extração, cifração e decifração. Os dois primeiros algoritmos, configuração e extração, são detalhados no capítulo 5, que discute o gerenciamento de chaves, pois estão relacionados à inicialização do sistema e a emissão das chaves privadas. Como o SEMAN possui uma abordagem distribuída, esses algoritmos possuem características particulares ao seu funcionamento.

A seguir são apresentados os algoritmos de cifração e decifração de mensagens. Na apresentação dos algoritmos, considera-se $k \in \mathbb{Z}^+$ como o parâmetro de segurança dado ao algoritmo de segurança e $\mathbb{G}$ como algum gerador de parâmetro BDH.

a. **cifração:** para cifrar $M$ usando a chave pública do nó $i$ os seguinte passos devem ser realizados:

   1) calcular $PK_i = H_1(N_i)$;

   2) escolher $r \in \mathbb{Z}_q^*$ aleatoriamente; e

   3) gerar o texto cifrado $C = \langle rP, M \oplus H_2(g_i^r) \rangle$ em que $g_i = \hat{e}(N_i, PK_i) \in \mathbb{G}_2^*$.

b. **decifração:** considera-se $C = \langle U, V \rangle$ o texto cifrado usando a chave pública do nó $i$. Para decifrar a mensagem é necessária a chave privada $SK_i$. A operação a seguir mostra como o texto pode ser decifrado:

$$V \oplus H_2(\hat{e}(SK_i, U)) = M$$

A prova do funcionamento desses algoritmos e a resistência a ataques pode ser encontrada em (BONEH; FRANKLIN, 2001).

#### 3.5.1.3 Criptografia baseada em identidade

Em 1984, Adi Shamir apresentou um novo modelo de criptografia assimétrica, chamado de IBC (SHAMIR, 1985), como alternativa para simplificar o gerenciamento de chaves públicas e certificados em uma *Publick Key Infrastructure* (PKI). Um IBC permite que qualquer par de usuários se comuniquem, de forma segura, e verifiquem mutuamente suas assinaturas sem a troca de chaves públicas e privadas, sem manter um diretório de chaves e sem usar os serviços de uma terceira entidade (ZHAO et al., 2012). Assim, o IBC visa a evitar o alto custo do gerenciamento de chaves públicas e autenticação de assinaturas presente em uma PKI tradicional. Contudo, a proposta de Shamir não apresentou soluções práticas para fornecer um esquema de *Identity-Based Encryption* (IBE). Apenas em 2001, Boneh e Franklim (BONEH; FRANKLIN, 2001) apresentaram o primeiro esquema IBE prático e seguro usando mapas bilineares. Esse esquema é conhecido como BF-IBE. Após esse estudo, outros esquemas baseados em identidade foram propostos, como o IBE hierárquico, *Identity-Based Signature* (IBS), autenticação baseada em identidade e protocolos de acordo de chaves. Sem a perda da generalidade, esta tese parte do esquema BF-IBE, embora outro esquema possa ser empregado na construção dos algoritmos.

Em um IBC, em vez de gerar um par aleatório de chaves pública e privada, um usuário escolhe uma *string* arbitrária, como o seu e-mail ou endereço de IP, para ser a sua chave pública. Assim, esse modelo de sistema elimina a necessidade de certificados de chave

pública e da propagação dos certificados e chaves públicas dos usuários pela rede. Então, um emissor pode cifrar uma mensagem para um receptor conhecendo apenas a identidade do receptor, sem precisar de um certificado de chave pública. Por outro lado, um usuário não pode emitir a sua própria chave privada. Para isso, é necessária uma entidade confiável para emitir as chaves privadas dos usuários, chamada de PKG. Este PKG é responsável também pela configuração do sistema e pela geração da chave mestre da rede.

A Figura B.1 apresenta uma exemplo do funcionamento de um IBC. Nesse exemplo, Beto envia uma mensagem para Ana. Ele utiliza a identidade conhecida de Ana associada à chave pública mestre do sistema para cifrar a mensagem que é transmitida. A chave de decifração é solicitada por Ana e gerada pelo PKG. Como um PKG emite todas as chaves privadas dos usuários, ele pode decifrar todas as mensagens desse usuário. Isso acontece porque o PKG detém a chave privada mestre. Esse problema é conhecido como custódia da chave[1]. Assim, os IBCs requerem que o PKG seja totalmente confiável, o que dificulta a sua implementação em ambientes dinâmicos, como as MANETs.

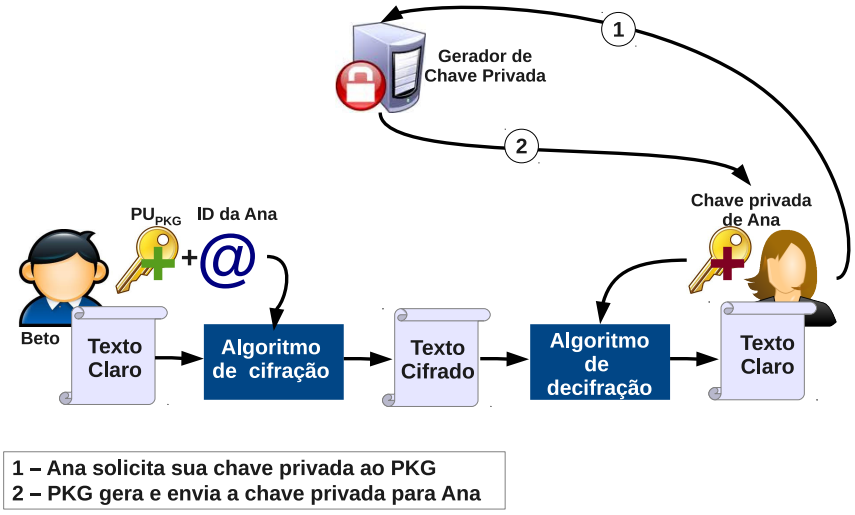



Figura 3.8: Visão geral do funcionamento dos criptossistemas baseados em identidade

De modo geral, os esquemas criptográficos baseados em identidade consideram quatro algoritmos: configuração, extração, cifração e decifração. Uma breve descrição de cada um desses algoritmos é apresentada a seguir:

a. "configuração": mapeia *strings* arbitrárias (identidade) para pontos em uma curva

[1]Comumente encontrado na literatura como *key escrow*.

elíptica. Configura a chave pública do sistema $PU_{PKG}$ como $sP$, em que $s$ é um número aleatório em $\mathbb{Z}_q^*$ e $P$ é um ponto arbitrário em $E/\mathbb{F}_p$ de ordem $q$. Escolhe uma função *hash* $H : \mathbb{F}_{p^2} \rightarrow \{0,1\}^n$ para algum $n$. Escolhe uma segunda função *hash* $G : \{0,1\}^* \rightarrow \mathbb{F}_p$. Os parâmetros do sistema são publicados como $\langle p, n, P, PU_{PKG}, G, H\rangle$. A chave mestre privada é $s \in \mathbb{Z}_q$;

b. "extração": para uma dada *string* $ID \in \{0,1\}^*$, o algoritmo constrói a chave pública para $ID : Q.ID = G(ID)$, um ponto em $E/\mathbb{F}_q$ mapeado a partir de $ID$, e chave privada $d.ID = s.Q.ID$;

c. "cifração": escolhe aleatoriamente $r \in \mathbb{Z}_q$ e gera um texto cifrado $C = rP, M \oplus H(g.ID)$ em que $g.ID = \hat{e}(Q.ID, PU_{PKG}) \in \mathbb{F}_{p^2}$; e

d. "decifração": Sendo $C = \langle U, V\rangle$ um texto cifrado usando a chave pública de $ID$, o algoritmo decifra $C$ usando a chave privada $d.ID : V \oplus H(\hat{e}(d.ID, U)) = M$.

### 3.5.2 Gerenciamento de Confiança

Este componente é responsável por fornecer informações para permitir que os nós estabeleçam conexões com níveis pré-determinados de confiança entre eles. Os valores de confiança que são calculados e disponibilizados pelo gerenciamento de confiança devem ser usados pelo SEMAN como suporte aos serviços fornecidos por ele. Os outros componentes do módulo de segurança, por exemplo, utilizam o gerenciamento de confiança para decidir se um serviço fornecido por outro nó pode ser considerado confiável ou não.

O gerenciamento de confiança auxilia o *middleware* a proteger a rede contra o mau comportamento dos nós. Por exemplo, os nós egoístas ou aqueles que apresentam um comportamento bizantino, podem ser isolado pelos demais nós com base nas informações obtidas deste componente. Também, o próprio gerenciamento de confiança foi projetado visando a mitigar o impacto dos ataques de falsa acusação (ou *bad mouthing*).

As informações de confiança também são disponibilizadas a outros módulos e componentes do SEMAN, que podem usá-las em suas operações internas. Por exemplo, o componente de armazenamento distribuído pode escolher servidores de armazenamento

mais confiáveis para solicitar um dado. Em outro exemplo, o componente de gerenciamento de recursos pode classificar os nós que estão hospedando um serviço tomando como base os valores de confiança. Essa classificação dos nós pode ser utilizada como suporte na decisão da escolha dos nós para os quais os pedidos serão encaminhados.

O esquema de gerenciamento de confiança empregado pelo *middleware* considera as observações diretas e indiretas (recomendações) dos nós. A troca de recomendações é realizada pelos nós que participam de um mesmo grupo de contexto. Assim, todas as informações e valores de confiança estão relacionados a um único contexto, que depende da aplicação que está sendo fornecida, do nó que está hospedando tal aplicação e do cenário em que ela está disponibilizada. Esta tese apresenta duas formas de se implementar o gerenciamento de confiança no SEMAN, que são discutidas e apresentadas no capítulo 4.

### 3.5.3 Gerenciamento de chaves

O gerenciamento de chaves consiste na administração segura das chaves criptográficas (MENEZES; OORSCHOT; VANSTONE, 1996). Ele deve considerar a geração, armazenamento, distribuição, proteção e revogação das chaves, e também garantir a disponibilidade aos nós autênticos. Nas MANETs, o gerenciamento de chaves deve tratar a topologia dinâmica e ser auto-organizado e descentralizado (SILVA et al., 2008). Além disso, deve considerar ameaças como o comprometimento da confidencialidade e da autenticidade das chaves públicas e privadas e o uso não autorizado dessas chaves (STALLINGS, 2009).

Também, um esquema robusto de gerenciamento de chaves para MANETs precisa satisfazer requisitos básicos como (MENEZES; OORSCHOT; VANSTONE, 1996): não ter um ponto único de falha; tolerância a comprometimentos; capacidade de revogar as chaves dos nós comprometidos e atualizar as chaves dos nós não-comprometidos; ser eficiente quanto ao armazenamento, o processamento e a comunicação. Com essas características, o gerenciamento de chaves fornece ao sistema a segurança contra os ataques que comprometem a confiabilidade das identidades e/ou chaves criptográficas apresentadas pelos nós, como os ataques de personificação e *Sybil*.

A arquitetura de um esquema de gerenciamento de chaves depende das técnicas criptográficas empregadas pelo sistema. Como o SEMAN considera o uso de criptossistemas baseados em identidade, o componente de gerenciamento de chaves também precisa ser baseado em identidade. Uma característica dos esquemas baseados em identidade é que a chave privada de todos os nós deve ser conhecida pelo PKG. Isto implica em dois desafios na implementação de um esquema de gerenciamento de chaves: a custódia das chaves e a disponibilidade do PKG. Dessa forma, é preciso uma arquitetura robusta de distribuição das atividades do PKG para garantir a confiabilidade do esquema.

O gerenciamento de chaves também deve fornecer meios para a realização do acordo de chaves, em que dois ou mais nós derivam uma chave de sessão comum, válida por um tempo limitado. Um serviço de acordo de chaves aplicável para as MANETs deve prevenir nós maliciosos de quebrar uma chave acordada entre os nós. Além disso, o protocolo não deve revelar a chave privada, derivada da chave pública, a qualquer nó malicioso.

Este protocolo de acordo de chaves deve ser leve e não envolver uma terceira autoridade confiável. Para reduzir o custo de comunicação e processamento, o SEMAN suporta o uso de acordo de chaves em todas as sessões longas de comunicação. Assim, uma vez que a chave de sessão é derivada, os nós comunicantes não precisam verificar a autenticidade dos outros nós por meio de operações criptográficas complexas e custosas. A implementação do gerenciamento de chaves no SEMAN é discutida no capítulo 5.

### 3.5.4 Gerenciamento de Grupos

Este componente mantém todas as informações sobre os grupos de contexto de que o nó seja membro. Ele é responsável por fornecer primitivas de inicialização dos grupos, entrada e saída de nós, descoberta de nós e gerenciamento dos contextos.

O módulo de gerenciamento de grupos é fundamental para o funcionamento do SEMAN. Como todos os serviços fornecidos são baseados em grupos de contexto, a administração destes grupos tem um papel muito importante para a eficácia e confiabilidade destes serviços.

Usando uma abordagem em grupos, unida à estratégia de compartilhamento do se-

gredo $t$-sobre-$n$, este componente aumenta a resistência do sistema contra ataques de egoísmo e ataques bizantinos. Contra os ataques de egoísmo pois as atividades de um grupo não dependem apenas de um nó. Assim, mesmo que alguns nós tenham um comportamento egoísta, a presença de $t << n$ nós bem comportados já possibilita a realização das atividades. Também contra os ataques bizantino, pois um atacante sozinho não tem condições de tomar decisões em nome dos demais membros do grupo. Ele precisaria do apoio de, pelo menos, outros $t$ membros de um grupo para poder comprometer o funcionamento do sistema.

Além de todo o gerenciamento dos grupos de contexto, este componente fornece o serviço para comunicação segura entre os membros de um grupo de contexto e a execução de aplicações distribuídas neste grupo. A comunicação entre os membros dos grupos de contexto deve ser protegida por algum método de criptografia. Com isso, o SEMAN garante que apenas membros legítimos de um grupo de contexto podem acessar os dados enviados para este grupo, mitigando o impacto dos ataques de personificação e *Sybil*. O gerenciamento de grupos é descrito no capítulo 6.

### 3.5.5 Gerenciamento de políticas

Para fornecer resistência a ataques maliciosos, o SEMAN integra todos esses componentes: gerenciamento de confiança, gerenciamento de chaves e gerenciamento de grupos. Essa integração é suportada por um conjunto de políticas adaptativas que são configuradas no *middleware*. Para isso, todos os componentes podem utilizar os serviços de um componente de gerenciamento de políticas integrado ao *middleware*. Esse componente gerencia as regras de acesso, privacidade, segurança e colaboração dos recursos do sistema. Ele inclui a definição de regras para interação, acesso a recursos, papéis e relacionamentos, e fornece ações que podem ser executadas de acordo com o comportamento do ambiente e os requisitos de segurança das aplicações.

A figura 3.9 ilustra as três funções básicas deste serviço: administração, cumprimento e manutenção das políticas. A administração das políticas é responsável por manter as restrições dos diferentes tipos de nós no uso do *middleware* e os papéis dos nós nos

relacionamentos entre as aplicações.

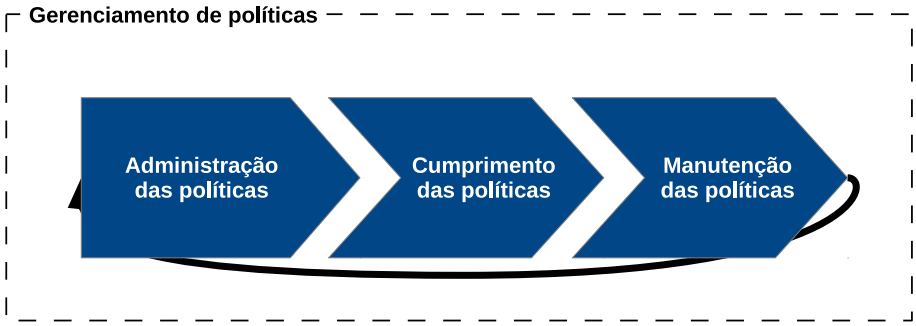



Figura 3.9: Funções do gerenciamento de políticas.

O cumprimento das políticas garante que todas as políticas de acesso, privacidade, segurança e colaboração definidas pelas aplicações serão suportadas e aplicadas pelo *middleware*. Ele fornece informações ao componente de gerenciamento de Autenticação, Autorização e Contabilização para que seja realizada a autenticação e autorização das aplicações com base nas políticas pré-definidas. Também fornece tais informações aos serviços fornecidos pelo *middleware* para garantir que os serviços troquem mensagens apenas com outras aplicações que respeitem as requisitos de segurança previamente estabelecidas. Por fim, a manutenção das políticas é responsável pela atualização e redefinição das regras e papéis previamente definidos. Sua função é receber informações das aplicações referentes às alterações em suas requisitos de segurança, bem como tratar as informações recebidas do ambiente externo. Com base nestas informações o gerenciamento de políticas pode aplicar regras mais rígidas, para garantir a segurança solicitada pelas aplicações.

Está fora do escopo desta tese determinar como esses valores serão definidos. Apenas, salienta-se que todos os parâmetros e variáveis, que são utilizados pelos componentes de segurança, podem ser armazenados e mantidos pelo serviço de gerenciamento de políticas.

## 3.6 Integração dos módulos e componentes

Para garantir a eficácia na oferta dos serviços às aplicações, é preciso que todos os módulos e componentes do *middleware* estejam integrados. A figura 3.10 ilustra as atividades básicas do SEMAN e como elas estão integradas no fornecimento de serviços às

aplicações. É importante ressaltar que esta figura não ilustra todas as atividades que podem ser fornecidas pelo *middleware*.

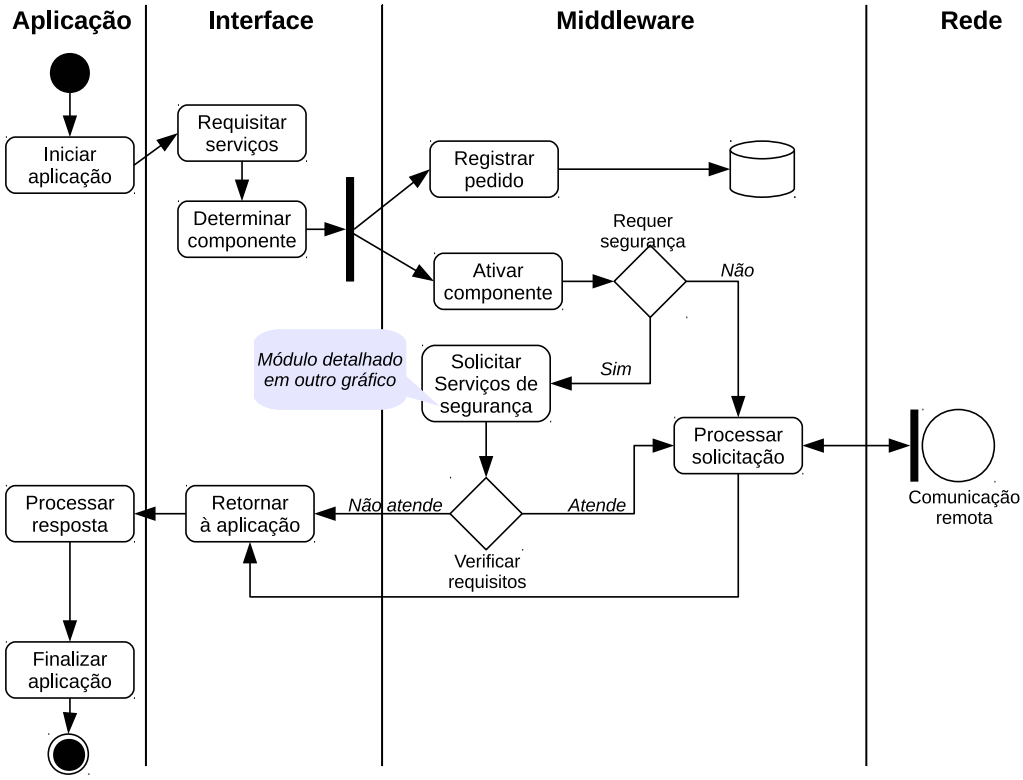



Figura 3.10: Atividades básicas do SEMAN.

Quando uma aplicação deseja utilizar os serviços do *middleware* ela realiza uma chamada de procedimentos ao SEMAN informando o serviço desejado e o seus parâmetros. Com base nessas informações, a Interface de Comunicação determina quais componentes são necessários para o fornecimento do serviço. Em paralelo, a Interface de Comunicação também acessa o componente de Registro de Pedidos para armazenar as informações relativas ao serviço e garantir a persistência da comunicação.

Então, os componentes internos do *middleware*, envolvidos no fornecimento do serviço, determinam, com base nos parâmetros informados, se a aplicação necessita dos serviços do módulo de segurança. Se esses serviços não forem necessários, o componente inicia o processamento do serviço. Caso contrário, os componentes de segurança são ativados e as aplicações e nós envolvidos na comunicação devem ser autenticados e autorizados, com base nas políticas de segurança. Se todos os requisitos de segurança forem atendidos, os componentes podem iniciar o processamento do serviço.

Na fase de processamento podem ser necessárias trocas de mensagens pela rede, com as outras partes comunicantes envolvidas no fornecimento do serviço desejado. Para isso, o *middleware* utiliza os serviços das camadas de rede subjacentes, sempre considerando os requisitos de segurança das aplicações.

Caso os serviços do módulo de segurança sejam necessários, os componentes envolvidos buscam as informações necessárias no objeto de armazenamento persistente, ou diretamente dos componentes de segurança. Para que as informações de segurança estejam sempre disponíveis e corretas é importante que os componentes envolvidos nessa atividade também estejam bem integrados, como ilustra a figura 3.11.

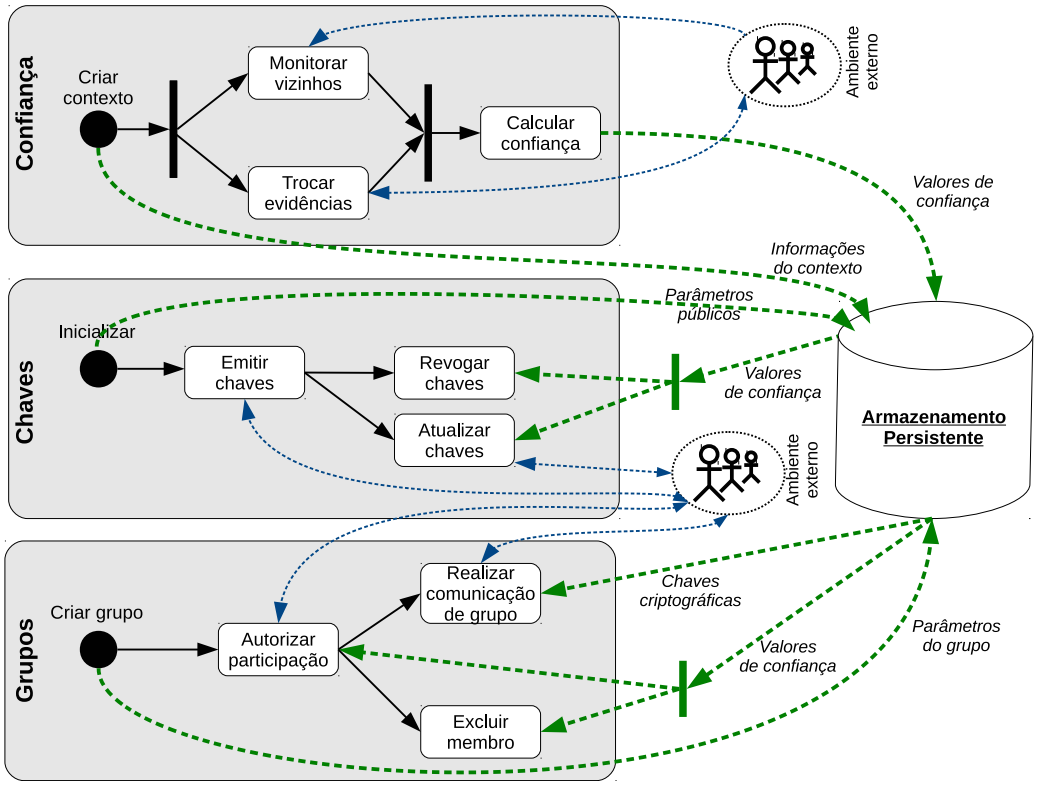



Figura 3.11: Atividades básicas do módulo de segurança.

É importante notar que todos os componentes de segurança interagem com o ambiente externo, trocando informações com os nós ou recebendo e solicitando pedidos, por exemplo. Além disso, todas as informações essenciais para o funcionamento seguro do *middleware* são enviadas para o objeto de armazenamento persistente.

Esse objeto de armazenamento persistente é útil também a comunicação assíncrona entre os componentes de segurança. Por exemplo, os valores de confiança dos nós, calcu-

lados pelo gerenciamento de confiança dentro de um contexto, são armazenados e podem ser utilizados pelo gerenciamento de chaves na tomada de decisão para atualização ou revogação das chaves criptográficas. Por fim, todas as tomadas de decisão, tanto as internas do módulo de segurança como aqueles dos demais serviços do *middleware*, são realizadas com base nas políticas de segurança pré-determinadas pelas aplicações.

## 3.7 Conclusão

Este capítulo apresentou a arquitetura de funcionamento do SEMAN. Foi discutido o funcionamento básico de dois módulos do *middleware*: de serviços e de processamento. No módulo de serviço foram apresentadas os principais componentes e serviços que são fornecidos pelo SEMAN e como eles podem usar as funcionalidades dos grupos de contexto no provimento destes serviços. No módulo de processamento foram discutidas as formas que o SEMAN pode fornecer o gerenciamento dos pedidos, de serviços e de componentes.

Com um maior detalhamento, foi apresentada uma visão geral do módulo de segurança, que é o ponto central do *middleware* proposto. Foram discutidos os componentes desse módulo e como eles podem ser integrados usando políticas pré-determinadas. Os próximos capítulos apresentam, detalhadamente, o funcionamento de cada um dos componentes do módulo de segurança.

# CAPÍTULO 4

# GERENCIAMENTO DE CONFIANÇA

Embora a criptografia possa ser usada para garantir a comunicação segura nas MANETs, ela não fornece informações sobre a confiabilidade dos nós (LI; SLAY; YU, 2005). Além disso, muitos mecanismos criptográficos, como o gerenciamento de chaves (LIMA et al., 2009; MERWE; DAWOUD; MCDONALD, 2007), dependem de algum grau de confiança pré-estabelecida entre os nós. Contudo, avaliar a confiança em qualquer tipo de rede aberta é muito difícil e é um tópico que tem recebido grande atenção da comunidade de segurança (BLAZE; FEIGENBAUM; LACY, 1996).

*Confiança* é um conceito das ciências sociais (LEWIS; WEIGERT, 1985), e pode ser definido como "a confiabilidade que um *outorgante* tem, ou o quanto ele está disposto a assumir de risco, em um *administrador*" (BUSKENS, 2002). Nesse contexto, o gerenciamento de confiança pode ser definido como um mecanismo para permitir que os nós, sem qualquer interação anterior, estabeleçam conexões com um nível pré-determinado de confiança entre si (BLAZE; FEIGENBAUM; KEROMYTIS, 1999). Exemplos de uso do gerenciamento de confiança incluem suporte em decisões como detecção de intrusões (ALBERS et al., 2002), autenticação (GHOSH; PISSINOU; MAKKI, 2005), controle de acesso (LUO et al., 2004), e isolamento de nós malcomportados em protocolos de roteamento (MARTI et al., 2000).

O uso de técnicas de avaliação de confiança para minimizar as ameaças de segurança é muito relevante em redes abertas (BETH; BORCHERDING; KLEIN, 1994). Nas MANETs, a confiança pode ser empregada em estratégias de roteamento, armazenamento distribuído, gerenciamento de localização e gerenciamento ou estabelecimento de chaves. Embora os esquemas de avaliação de confiança sejam essenciais para muitos serviços de segurança, a maioria dos esquemas encontrados na literatura ou não consideram ou não são avaliados sob ataques maliciosos. Além disso, os poucos esquemas que consideram a

presença de nós maliciosos estão limitados a uma única operação da rede, como o roteamento.

Devido às características das MANETs, alguns conceitos e características devem ser cuidadosamente definidos (GOLBECK, 2006; SUN et al., 2006b), a saber:

a. *confiança não é necessariamente transitiva*: se o nó $N_a$ confia no nó $N_b$, e o nó $N_b$ onfia no nó $N_c$, não é verdade que o nó $N_a$ confia no nó $N_c$, mas isso pode ser considerado;

b. *confiança é assimétrica*: o fato do nó $N_a$ confiar no nó $N_b$ não significa necessariamente que o nó $N_b$ também confia no nó $N_a$;

c. *confiança é subjetiva*: como a confiança é uma herança da opinião pessoal, dois nós podem avaliar de forma diferente a confiabilidade de um outro nó;

d. *confiança é dependente de contexto*: o nó $N_a$ pode confiar no nó $N_b$ quando este está fornecendo serviço de roteamento mas não quando ele está fornecendo um outro serviço;

e. *avaliação de confiança deveria ser totalmente distribuída*: os esquemas não devem confiar em uma terceira entidade para determinar a confiança dos nós;

f. *o gerenciamento de confiança deveria considerar nós não-cooperativos*: ambientes com restrição de recursos, como as MANETs, são compostos por nós que podem apresentar um comportamento egoísta;

g. *valores de confiança devem ser contínuos*: o nível de confiança em um nó deve ser medido usando valores reais e contínuos; e

h. *confiança é dinâmica*: como os valores de confiança representam uma opinião pessoal, os nós podem alterar a sua avaliação sobre outros nós.

Nesta tese são apresentadas duas propostas para avaliar a confiança entre os pares comunicantes em uma rede. A primeira abordagem, chamada de *TRUst Evaluation service for MANETs* (TRUE), utiliza o conceito de cadeias de confiança que são formadas entre

os nós a partir de monitoramento direto e recomendações de vizinhos físicos. A segunda abordagem, chamada de *Trust with UltimatuM game* (TrustUM), emprega o Jogo do Ultimato, uma técnica da teoria dos jogos, no gerenciamento e avaliação da confiança entre os nós.

As duas abordagens propostas são compatíveis com os serviços fornecidos pelo SEMAN. A primeira é indicada para ambientes menos hostís, pois possui uma maior dependência na transitividade da confiança. Contudo, dependendo dos parâmetros utilizados, o efeito da transitividade pode ser mitigado. Por outro lado, a segunda abordagem emprega uma solução mais robusta nas trocas de informações, ou recomendações, baseada em teoria de jogos. Contudo, apresenta uma sobrecarga maior de processamento, devido aos cálculos que precisam ser realizados nas tomadas de decisão. Entretanto, as duas abordagens são resistentes a ataques maliciosos de propagação de informações falsas pela rede.

A próxima seção apresenta os trabalhos relacionados ao gerenciamento de confiança. Em seguida, as outras seções apresentam as duas abordagens, discutindo as suas características de funcionamento. Além disso, a eficácia das duas abordagens é avaliada por meio de simulações que são descritas ao longo das seções. Nas simulações, foram considerados cenários sem ataques e cenários com nós maliciosos, que realizam ataques de propagação de informações falsas. Esse tipo de ataque é conhecido como *bad mouthing*. As descrições e avaliações desse capítulo consideram a notação apresentada na Tabela 4.1.

## 4.1 Trabalhos relacionados

Muitos esquemas de avaliação de confiança foram proposto a fim de suportar ou manter as evidências de segurança dos nós nas MANETs. Em (JIANG; BARAS, 2004), é proposto o *Ant-Based Evidence Distribution* (ABED), baseado em inteligência coletiva, que afirma ser altamente distribuído e adaptativo à mobilidade dos nós. No ABED, os nós interagem entre si por meio de agentes ("*ants*" - formigas), que são capazes de identificar um caminho ótimo para acumular evidência de confiança. Contudo, ele não foi avaliado sob nenhum tipo de ataque.

Tabela 4.1: Notação utilizada.

| Notação | Descrição |
|---|---|
| $N_i$ | identidade do nó $i$ |
| $TV_{(N_x,N_v)}$ | valor de confiança do nó $N_x$ no nó $N_v$ |
| $TC^{x}_{(N_x,N_v)}$ | cadeia de confiança $x$ do nó $N_x$ para o nó $N_v$ |
| $a\|\|b$ | informação $a$ concatenada com informação $b$ |
| $G_{tr}$ | grafo da rede confiança baseada em contexto |
| $G^{x}_{tr}$ | grafo da rede confiança baseada em contexto do nó $N_x$ |
| $\|Z\|$ | tamanho de um dado conjunto $Z$ |
| $N_a \rightarrow N_b$ | nó $N_a$ confia no nó $N_b$ |
| $\Delta_T$ | intervalo entre as atividades de monitoramento de vizinhos |
| $\Delta T_{ex}$ | intervalo entre as trocas de informação |
| $\alpha$ | limite das trocas de informação |
| $\beta$ | limite das avaliações de confiança |
| $\cong$ | aproximadamente |

Em (THEODORAKOPOULOS; BARAS, 2006), um esquema de avaliação de evidência de confiança é proposto. Ele é modelado como um problema de caminho em um grafo direcionado. Esse esquema considera um nó de origem como uma entidade confiável para suportar a infraestrutura, violando as característica descentralizadas das MANETs. Além disso, os valores de confiança são representados de forma binária.

Um domínio físico-lógico auto-organizado baseado em confiança para o agrupamento de nós e suporte ao controle distribuído na rede é apresentado em (VIRENDRA et al., 2005). Ele introduz uma arquitetura de segurança baseada em domínio de confiança que usa a confiança para estabelecer chaves simétricas entre os nós de um grupo. Embora os autores descrevam a formalização e avaliação de confiança, o esquema não foi avaliado sob ataques e é aplicável apenas para o estabelecimento de chaves de grupos.

Uma avaliação de reputação distribuída que afirma prevenir a entrada de nós maliciosos em uma comunidade confiável foi proposto em (BOUKERCHE; REN, 2008). Contudo, nenhum modelo de ataques específicos foi considerado. Em (ZUO; HU; O'KEEFE, 2009), um algoritmo de cálculo de confiança foi proposto, para avaliar a confiança usando um

grafo de certificados de confiança. Contudo, o uso de certificados de segurança implica na verificação de assinaturas digitais com um nó ou entidade confiável.

Os esquemas de avaliação de confiança também têm sido empregados para suportar outras aplicações em MANETs, tais como autenticação e roteamento de pacotes. Em (CHANG et al., 2009), por exemplo, é proposto um esquema de avaliação de confiança para suportar a autenticação segura em MANETs. Ele assume que os nós formam grupos usando servidores de autoridade certificadora primários e secundários dentro da rede. Os valores de confiança dos nós aumentam ou diminuem baseados em seus valores prévios usando um modelo de confiança de cadeias de Markov. Então, o nó com o maior valor de confiança é selecionado como servidor da autoridade certificadora, e o nó com o segundo maior valor é o servidor secundário. Contudo, o esquema cria uma autoridade certificadora centralizada, que não é desejável nas MANETs.

Em (HE; WU; KHOSLA, 2004), é apresentado o SORI, que emprega o incentivo a cooperação baseado em reputação, estimulando o encaminhamento de pacotes e disciplinando o egoísmo por meio de punições. No SORI, a reputação do nó é calculada usando métricas objetivas, como a efetividade no encaminhamento de pacotes. Contudo, ele considera que a reputação de um nó é útil somente para os vizinhos físicos desse nó. Essa característica torna a implementação do SORI para suportar outras aplicações muito difícil. Outros esquemas que usam reputação ou estimativa de confiança para estimular o roteamento de pacotes podem ser encontrados na literatura (BUCHEGGER; BOUDEC, 2002b; DAI; JIA; QIN, 2009; MICHIARDI; MOLVA, 2002). Contudo, nenhum deles foi avaliado sob ataques e são limitados ao suporte de estratégias de roteamento.

Em (VELLOSO et al., 2008) é apresentado um modelo de confiança que afirma ser resistente a ataques *slander* (calúnia), uma variação do ataque *bad mouthing*. Esse esquema fornece nós com um mecanismo para construir um relacionamento de confiança com seus vizinhos. Contudo, o esquema permite que os nós avaliem apenas a confiança dos vizinhos físicos. Assim, a solução não é aplicável para aplicações que requerem informações de confiança de nós que estão foram do raio de alcance. Em (SUN et al., 2006a) é apresentado um esquema de avaliação de confiança que considera ataques maliciosos.

Contudo, ele é projetado apenas para operações de roteamento seguro, e detecta somente nós maliciosos atuando nos protocolos de roteamento.

## 4.2 TRUE: Serviço de avaliação de confiança

Essa seção descreve a primeira abordagem para avaliação de confiança para MANETs, o TRUE, publicado em (SILVA; MISAGHI; ALBINI, 2012a) e (SILVA; MISAGHI; ALBINI, 2012b). O TRUE suporta aplicações de forma dinâmica e autônoma, enquanto mantém a capacidade de resistir a ataques maliciosos. Nessa abordagem, cada nó cria, auto-organizadamente, uma rede de confiança baseada em contexto, para fornecer informações de confiança representada por um grafo direto $G_{tr} = (V_{tr}, E_{tr})$, em que os vértices $V_{tr}$ são os nós e as arestas $E_{tr}$ são as relações de confiança entre eles. Ela contém todas as informações de confiança que um nó tem sobre outros nós dentro de um contexto. Essas informações, ou evidências, são obtidas via interação direta ou via recomendação, considerando as políticas de segurança do sistema. A confiança de um nó é sempre calculada localmente, sem qualquer tipo de troca de mensagem, baseada na rede de confiança do nó.

Nas próximas seções, é apresentado como os nós criam seus grafos de redes de confiança baseadas em contexto e como ele pode atualizar esses grafos, obtendo evidências, ou recomendações, de outros nós. Em seguida, é descrito como os nós avaliam os valores de confiança de outros nós e como eles podem integrar as informações de diferentes nós. Por fim, é apresentada uma avaliação da eficácia do esquema em cenários sem ataques e em cenários com ataques maliciosos.

### 4.2.1 Construindo redes de confiança baseada em contexto

Quando está entrando no sistema, cada nó cria sua própria rede de confiança $G^i_{tr} = (V^i_{tr}, E^i_{tr})$ auto-organizadamente. Inicialmente, os nós possuem informações apenas sobre os nós com os quais tiveram relações de confiança direta, e somente esses dados são armazenados na rede de confiança. Então, em intervalos de tempo pré-determinado ($\Delta T_{ex}$),

os nós trocam, com seus vizinhos físicos, as evidências de confiança armazenadas em suas redes de confiança. Assim, os valores de confiança rapidamente serão propagados pela rede, seguindo um comportamento epidêmico (MICKENS; NOBLE, 2005; ZHANG et al., 2007).

As trocas de informação de confiança ocorrem da seguinte forma:

a. em intervalos $\Delta T_{ex}$, cada nó $N_x$ cria uma Mensagem de Informação de Confiança, denotada por $TIM = [G^x_{tr} \| N_x \| timestamp]$. Essa mensagem contém todas as evidências de confiança armazenadas em sua rede de confiança baseada em contexto, sua identidade, e o carimbo de tempo;

b. após criar essa mensagem, o nó $N_x$ envia essa mensagem para todos os seus vizinhos;

c. ao receber uma mensagem TIM, o nó $N_v$ avalia a relevância das evidências recebidas calculando a confiabilidade do nó $N_x$ ($TV_{(N_v,N_x)}$). Então, ele decide se aceita ou não essas evidências, baseado em suas políticas locais. Para isso, cada nó tem um valor de limiar, $\alpha$, em que ele aceita as evidências de confiança se, e somente se $TV_{(N_v,N_x)} \geq \alpha$;

d. se $N_v$ aceita as evidências de confiança, ele incorpora as informações recebidas em sua rede de confiança baseada em contexto; e

e. caso contrário, as evidências de confiança são descartadas.

### 4.2.2 Avaliação de confiança

Para avaliar a confiança do nó $N_u$, o nó $N_x$ deve ter uma conexão direta com o nó $N_u$ em $G^x_{tr}$ ou deve encontrar pelo menos uma cadeia de confiança (TC) de $N_x$ para $N_u$ em $G^x_{tr}$. As cadeias de confiança representam a confiança transitiva de $N_x$ em $N_u$. O grafo de rede de confiança $G^x_{tr}$ é ilustrado na Figura 4.1. Como o nó $N_x$ pode encontrar diferentes cadeias de confiança distintas entre ele e $N_u$ em $G^x_{tr}$, cada cadeia é denotada como $TC^i_{(N_x,N_u)}$.

Se $N_x$ possui uma relação de confiança com $N_u$, apenas esse valor é considerado na avaliação de confiança. Considerando o exemplo da Figura 4.1, é possível notar que $N_x$ tem um relacionamento com o nó $N_q$, e ele tem 80% de confiança nos serviços fornecidos pelo nó $N_q$ nesse contexto. Contudo, nesse exemplo $N_x$ não tem uma relação de confiança direta com o nó $N_u$. Assim, ele tenta encontrar um caminho de confiança em $G^x_{tr}$, estimando a confiabilidade de cada cadeia e calcular uma média ponderada dessas cadeias para cada nó.

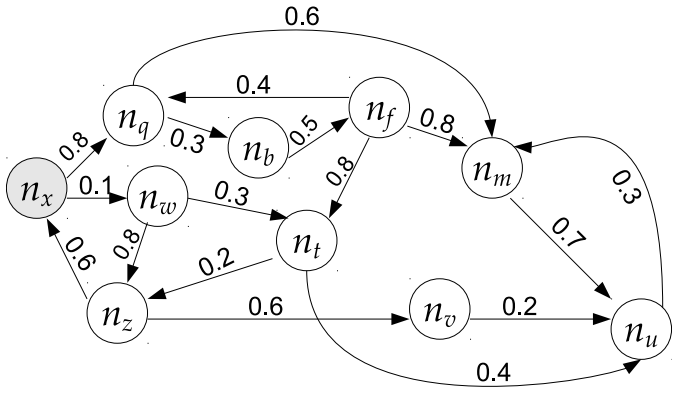



Figura 4.1: Exemplo da cadeia de confiança $G^x_{tr}$ do nó $N_x$.

Ao encontrar uma cadeia, o nó $N_x$ deve calcular sua confiança. Considerando que $N_1$ a $N_m$ sejam os $m$ nós intermediários na $i^a$ cadeia de confiança, denotada como $TC^i_{(N_x,N_u)}$, a equação 4.1 estima a confiabilidade de $TC^i_{(N_x,N_u)}$:

$$TC^i_{(N_x,N_u)} = TV_{(N_x,N_1)} \times \prod_{j=1}^{m-1} TV_{(N_j,N_{j+1})} \times TV_{(N_m,N_u)} \tag{4.1}$$

Retornando à Figura 4.1, existem diversas cadeias entre $N_x$ e $N_u$, por exemplo:

a. cadeia $(N_x \rightarrow N_q \rightarrow N_m \rightarrow N_u)$, com valor da cadeia de confiança $TC^1_{(N_x,N_u)} = 0,8 \times 0,6 \times 0,7 = 0,336$;

b. cadeia $(N_x \rightarrow N_q \rightarrow N_b \rightarrow N_f \rightarrow N_m \rightarrow N_u)$, com valor da cadeia de confiança $TC^2_{(N_x,N_u)} = 0,8 \times 0,3 \times 0,5 \times 0,8 \times 0,7 = 0,067$.

Além disso, os nós podem usar o valor de limiar para cada aresta da cadeia de confiança (valor $\beta$). Se no mínimo uma aresta da cadeia de confiança tem um valor de confiança

abaixo do limiar, a cadeia é desconsiderada. Por exemplo, se o nó $N_x$ considera $\beta > 0,4$, ele descartaria a cadeia 2 do exemplo anterior, já que ela possui uma aresta com valor de confiança igual a $0,3$.

Após calcular o valor de confiança de todas as cadeias, o valor de confiança $TV_{(N_x,N_u)}$ pode ser calculado aplicando uma média ponderada, como segue (equação 4.2):

$$TV_{(N_x,N_u)} = \frac{\sum_{i=1}^{k}(TC^i_{(N_x,N_u)} \times (1/|TC^i_{(N_x,N_u)}|))}{\sum_{i=1}^{k}\frac{1}{|TC^i_{(N_x,N_u)}|}} \tag{4.2}$$

A média ponderada reduz o impacto da transitividade nas cadeias de confiança, De fato, quanto maior a cadeia, menor é a sua confiabilidade. Assim, esse método visa a privilegiar as cadeias menores, seguindo uma perspectiva social.

### 4.2.3 Simulações e Resultados

O Network Simulator versão 2.34 foi utilizado para avaliar o desempenho e a eficácia do TRUE. As simulações foram realizadas considerando a presença de nós honestos e nós maliciosos. Os nós maliciosos alteram os valores de confiança de outros nós de forma não previsível e arbitrariamente com o objetivo de prejudicar o sistema.

Nas simulações, 100 nós usam o IEEE 802.11 com a função de coordenação distribuída como protocolo de controle de acesso ao meio. A propagação do sinal segue o modelo de reflexão no solo de dois raios e o raio de comunicação é de 120m. Os nós se movimentam em uma área de 1000 x 1000m, seguindo o modelo de mobilidade *waypoint* aleatório com velocidade máxima de 20 m/s e tempo de pausa de 20 s. O tempo total das simulações é de 2000s e os resultados são a média de 35 simulações com intervalo de confiança de 95%.

Durante a formação da rede, cada nó gera, aleatoriamente, os valores de confiança dos nós que ele confia. As relações de confiança inicial seguem uma distribuição de lei de potência, em que apenas poucos nós possuem muitas relações de confiança (no máximo 15). A distribuição de lei de potência aproxima corretamente a operação de confiança em redes dinâmicas, como as redes P2P e MANETs (RIPEANU; FOSTER; IAMNITCHI, 2002).

Então, os valores de confiança são configurado aleatoriamente seguindo uma distribuição normal de valores contínuos entre 0 e 1. O intervalo de trocas de informaçao $\Delta T_{ex}$ é de 10 segundos.

O TRUE foi avaliado sob três aspectos: (*i*) custo de comunicação; (*i*) média da confiança calculada nas redes de confiança e porcentual de nós considerados confiáveis em cenários sem ataques; (*iii*) média da confiança calculada nas redes de confiança e porcentual de nós que são considerados confiáveis em cenários com ataques de *bad mouthing*.

#### 4.2.3.1 Custo de comunicação

A sobrecarga de comunicação é muito pequena. O TRUE usa apenas mensagens a um salto de distância para atualizar as redes de confiança, e não utiliza mensagens adicionais para construir as cadeias de confiança, i.e. não precisa realizar trocas de mensagens para estimar a confiança dos outros nós. Assim, o custo de comunicação depende exclusivamente das mensagens de atualização.

Além disso, é possível aumentar $\Delta T_{ex}$ para reduzir o custo de comunicação. Essa função pode ser útil para adiar a exaustão de bateria de um nó. Contudo, o tempo para disseminar as evidências de confiança depende diretamente de $\Delta T_{ex}$. Um valor mais alto de $\Delta T_{ex}$ implica em um maior atraso para disseminar as evidências.

A sobrecarga de memória também é pequena. Os nós devem manter apenas as redes de confiança baseadas em contexto. Por outro lado a sobrecarga computacional para manter o esquema atualizado pode ser significativa. Os nós devem computar os valores de confiança de todas as mensagens TIM recebidas. Se o nó decide aceitar uma mensagem TIM, ele deve recalcular o grafo de rede confiança inteiro considerando as novas informações. Consequentemente, a sobrecarga computacional depende diretamente de $\Delta T_{ex}$ e do número de vizinhos de cada nó.

#### 4.2.3.2 Cenários sem ataques

Considerando os cenários sem atacantes, o TRUE foi avaliado variando o limite para as trocas de informações ($\alpha$) e o limite para os valores das cadeias de confiança ($\beta$). É

esperado que em cenários com limites mais rigorosos, os nós consigam obter informações de confiança sobre um conjunto menor de nós e, dessa forma, estimar a confiança de poucos nós.

A Figura 4.2 mostra a confiança média calculada nas rede de confiança baseadas em contexto. Os valores de confiança estimados também são representados na Tabela 4.2. Em cenários com $\alpha = 0,1$ e $\beta = 0,1$, o valor de confiança médio é aproxidamente $0,2$ pois o nó aceita recomendações de outros nós com valores de confiança pequenos. Nesse caso, as cadeias de confiança podem ser formadas com nós não confiáveis.

Tabela 4.2: Média dos valores de confiança estimados - Valores do gráfico.

| $\beta$ | $\alpha$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0,0 | 0,1 | 0,2 | 0,3 | 0,4 | 0,5 | 0,6 | 0,7 | 0,8 | 0,9 |
| 0,0 | 0,07 | 0,19 | 0,23 | 0,27 | 0,31 | 0,35 | 0,40 | 0,44 | 0,49 | 0,52 |
| 0,1 | 0,07 | 0,19 | 0,23 | 0,27 | 0,31 | 0,35 | 0,40 | 0,44 | 0,49 | 0,52 |
| 0,2 | 0,07 | 0,19 | 0,23 | 0,27 | 0,31 | 0,35 | 0,40 | 0,45 | 0,49 | 0,52 |
| 0,3 | 0,08 | 0,19 | 0,23 | 0,27 | 0,31 | 0,35 | 0,40 | 0,45 | 0,49 | 0,52 |
| 0,4 | 0,08 | 0,19 | 0,23 | 0,27 | 0,31 | 0,35 | 0,40 | 0,45 | 0,49 | 0,52 |
| 0,5 | 0,09 | 0,20 | 0,24 | 0,28 | 0,31 | 0,35 | 0,40 | 0,45 | 0,49 | 0,52 |
| 0,6 | 0,11 | 0,21 | 0,25 | 0,29 | 0,32 | 0,36 | 0,40 | 0,45 | 0,49 | 0,52 |
| 0,7 | 0,19 | 0,27 | 0,30 | 0,33 | 0,35 | 0,38 | 0,41 | 0,45 | 0,49 | 0,52 |
| 0,8 | 0,36 | 0,39 | 0,40 | 0,42 | 0,43 | 0,44 | 0,45 | 0,47 | 0,49 | 0,53 |
| 0,9 | 0,51 | 0,52 | 0,52 | 0,52 | 0,52 | 0,52 | 0,52 | 0,52 | 0,52 | 0,53 |

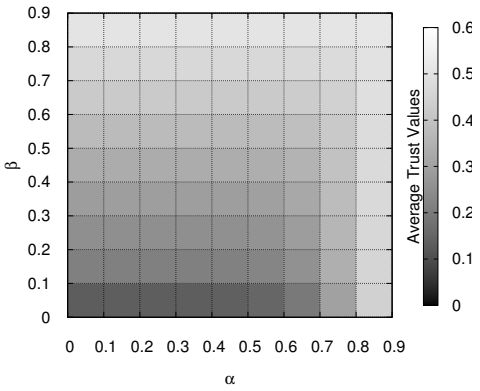



Figura 4.2: Média dos valores de confiança estimados.

Por outro lado, em cenários com $\alpha = 0,9$ and $\beta = 0,9$, o valor de confiança médio é $0,53$. É possível observar que os valores de $\alpha$ e $\beta$ impactam nos resultados. Com $\alpha = 0,6$, a média dos valores de confiança calculados é sempre maior que $0,4$, independente de $\beta$. É importante destacar que o objetivo desse esquema não é aumentar a confiabilidade, mas

estimá-la.

A Figura 4.3 mostra a porcentagem de nós que são considerados confiáveis para cada nó variando os valores de $\alpha$ e $\beta$. A porcentagem de nós confiáveis também é representada na Tabela 4.3. Esse resultado está diretamente relacionado com os resultados apresentados na Figura 4.2. Em cenário com $\alpha \leq 0,4$ e $\beta \leq 0,5$, a porcentagem de nós que são considerados confiáveis é maior que 95%. Se $\alpha = 0,8$, a porcentagem de nós confiáveis é cerca de 30%. Se $\alpha = 0,9$ e $\beta = 0.9$, essa porcentagem é próxima a 15%.

Tabela 4.3: Porcentagem de nós confiáveis sem atacantes - Valores do gráfico.

| $\beta$ | $\alpha$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0,0 | 0,1 | 0,2 | 0,3 | 0,4 | 0,5 | 0,6 | 0,7 | 0,8 | 0,9 |
| 0,0 | 99,00 | 99,00 | 99,00 | 98,79 | 97,06 | 90,48 | 75,61 | 52,76 | 31,11 | 15,70 |
| 0,1 | 99,00 | 99,00 | 99,00 | 98,79 | 97,06 | 90,48 | 75,61 | 52,76 | 31,11 | 15,70 |
| 0,2 | 99,00 | 99,00 | 99,00 | 98,79 | 97,06 | 90,48 | 75,58 | 52,73 | 31,09 | 15,68 |
| 0,3 | 98,93 | 98,95 | 98,96 | 98,76 | 96,99 | 90,41 | 75,54 | 52,66 | 31,06 | 15,67 |
| 0,4 | 98,48 | 98,54 | 98,51 | 98,33 | 96,71 | 90,23 | 75,33 | 52,56 | 31,01 | 15,64 |
| 0,5 | 96,99 | 97,07 | 97,05 | 96,92 | 95,64 | 89,60 | 74,93 | 52,28 | 30,83 | 15,54 |
| 0,6 | 92,62 | 92,73 | 92,85 | 92,75 | 91,68 | 86,84 | 73,82 | 51,71 | 30,42 | 15,32 |
| 0,7 | 79,23 | 79,59 | 79,92 | 79,95 | 79,09 | 75,53 | 66,24 | 49,56 | 29,53 | 14,99 |
| 0,8 | 50,71 | 50,78 | 50,94 | 51,07 | 51,09 | 50,22 | 46,84 | 38,90 | 27,90 | 14,71 |
| 0,9 | 17,41 | 17,41 | 17,42 | 17,42 | 17,42 | 17,44 | 17,44 | 17,41 | 16,94 | 14,58 |

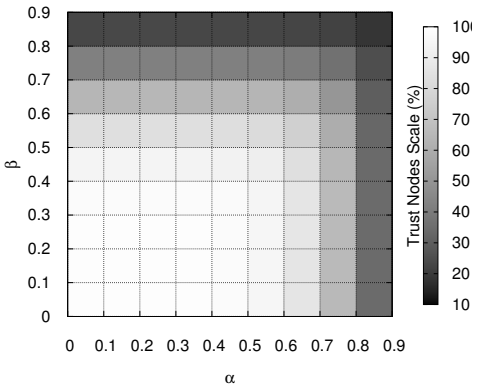



Figura 4.3: Porcentagem de nós confiáveis sem atacantes.

A Tabela 4.4 mostra o tempo médio (em segundos) necessário para propagar uma informação de confiança pela rede e o tamanho médio das redes de confiança baseadas em contexto, considerando o limite para trocas de informação ($\alpha$). Note que o tempo para disseminar a informação é menor com $\alpha = 0,0$ ou $\alpha = 0,9$ e é maior com $\alpha = 0,4$ e $\alpha = 0,5$. Isso ocorre porque com $\alpha = 0,0$ os nós aceitam evidências de confiança de

todos os demais nós da rede. Assim, os dados são trocados rapidamente. Num outro extremo, se $\alpha = 0,9$, os nós aceitam as evidências de confiança apenas dos amigos mais próximos. Nesse caso, mesmo que ele não tenha muitas informações armazenadas localmente, ele não aceitará informações de outros nós, também, aumentando o valor de $\alpha$, menos informações são trocadas e as redes de confiança são menores, i.e. poucos nós são armazenados localmente nas redes de confiança baseadas em contexto.

Tabela 4.4: Tempo para disseminar as evidências de confiança e porcentagem de nós nas redes de confiança

| $\alpha$ | **Tempo (seg.)** | **Nós (%)** |
|---|---|---|
| 0,0 | 198,51 | 100,00% |
| 0,1 | 713,54 | 99,99% |
| 0,2 | 801,49 | 99,96% |
| 0,3 | 885,14 | 99,92% |
| 0,4 | 936,97 | 99,35% |
| 0,5 | 926,96 | 96,94% |
| 0,6 | 878,08 | 92,57% |
| 0,7 | 768,51 | 78,98% |
| 0,8 | 598,22 | 50,37% |
| 0,9 | 281,08 | 17,34% |

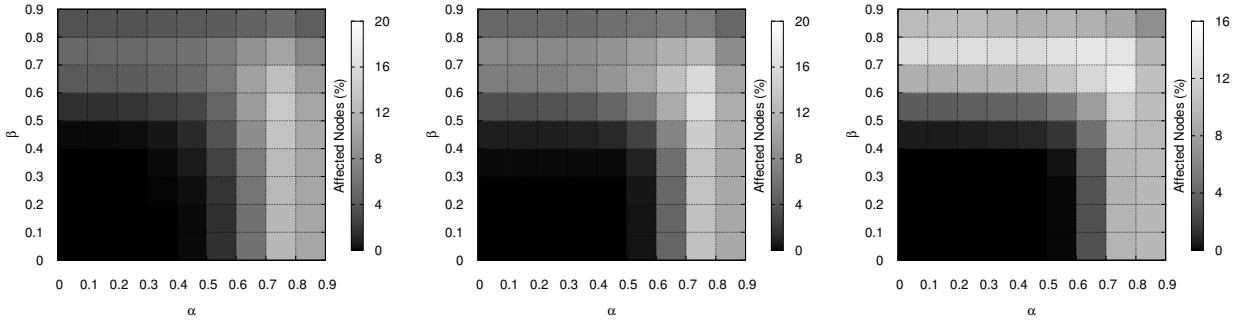
#### 4.2.3.3 Cenários com atacantes

O TRUE também foi avaliado com cenários sob ataques *bad mouthing*. Os ataques de *bad mouthing* consistem em nós maliciosos que fornecem evidências de confiança desonestas para difamar nós não comprometidos ou aumentar os valores de confiança de nós maliciosos (DELLAROCAS, 2000). Em todos os cenários, os ataques iniciam depois que os nós constroem suas redes de confiança.

O TRUE foi avaliado sob ataques de *bad mouthing*, em que os nós maliciosos alteram os valores de outros nós para 1,0. Também foi considerado que os nós maliciosos podem executar um ataque em conluio, em que diversos atacantes escolhem o mesmo nó para alterar o valor de confiança.

A Figura 4.4 mostra o impacto dos ataques. Note que em cenários com $\alpha$ e $\beta$ pequenos, a porcentagem de nós afetados é próxima de 0. Isso ocorre porque, nesses cenários, os nós corretos já confiam nos nós maliciosos, então o ataque não altera o comportamento

da rede. Os resultados mostram que o pior caso ocorre com $\alpha \cong 0,7$ e $\beta \cong 0,7$ e 10% de atacantes (Figura 4.4(c)). Nesse caso, a porcentagem de nós comprometidos é apenas 15%.




(a) 2% de atacantes (b) 5% de atacantes (c) 10% de atacantes

Figura 4.4: Cenários sob ataques de *bad mouthing*.

A Tabela 4.5 mostra quanto o sistema é afetado em cenários com atacantes, avaliando a variação dos valores de confiança calculados pelos nós. Essa avaliação considera apenas cenários com $\alpha = 0,0$, i.e cenários em que os nós trocam evidência de confiança com todos os outros nós. Note que diante de 5% de atacantes e $\beta = 0,9$, a variação dos valores de confiança é $0,3277$. Também, em cenários com 10% de atacantes e $\beta < 0,6$, a variação é sempre abaixo de $0,2$.

Tabela 4.5: Variação de confiança em cenário sob ataque

| $\beta$ | Atacantes | | |
|---|---|---|---|
| | **2%** | **5%** | **10%** |
| 0,0 | 0,0055 | 0,0116 | 0,0158 |
| 0,1 | 0,0271 | 0,0503 | 0,0671 |
| 0,2 | 0,0365 | 0,0655 | 0,0868 |
| 0,3 | 0,0432 | 0,0803 | 0,1045 |
| 0,4 | 0,0561 | 0,0987 | 0,1290 |
| 0,5 | 0,0884 | 0,1355 | 0,1700 |
| 0,6 | 0,1619 | 0,2081 | 0,2390 |
| 0,7 | 0,3258 | 0,3403 | 0,3574 |
| 0,8 | 0,4300 | 0,4403 | 0,4339 |
| 0,9 | 0,2829 | 0,3277 | 0,3377 |

## 4.3 TrustUm: Confiança usando o Jogo do Ultimato

Um outro exemplo de gerenciamento de confiança que pode ser integrado ao módulo de segurança do SEMAN, é o TrustUM (NICHELE, 2012). Ele estima a confiabilidade entre os nós considerando observações diretas ou recomendações de outros nós. Os nós trocam recomendações baseados na teoria de jogos, mais especificamente o Jogo do Ultimato. O Jogo do Ultimato é usado para modelar o relacionamento entre dois nós vizinhos e decidir como eles trocam recomendações entre si. Além disso, a confiança é sempre calculada localmente, sem qualquer troca de mensagem.

O TrustUM pode estimar a confiança que um nó tem em outro nó, fornecendo os valores de confiança para aplicações ou protocolos. A forma com que esses valores serão utilizados depende da aplicação ou do protocolo. A avaliação de confiança fornecida pelo TrustUM não depende de qualquer protocolo ou funcionalidade de segurança da rede. Assim, ele pode ser usado pelos protocolos de roteamento, mecanismos criptográficos, gerenciamento de chaves, ou qualquer aplicação que necessite de informação de confiança. Detalhes do funcionamento do TrustUM e a sua eficácia em cenários com ataques podem ser encontrados na sua dissertação original (NICHELE, 2012).

### 4.3.1 Jogo do Ultimato

O Jogo do Ultimato foi proposto em 1982 por Guth et al. (GUTH; SCHMITTBERGER; SCHWARZE, 1982). Nesse jogo, uma quantidade de dinheiro pré determinada (e.g. 100 reais) deve ser dividida entre dois jogadores $i$ e $j$ observando as seguintes regras:

a. o jogador $i$ propõe uma divisão ao jogador $j$. O jogador $i$ determina quando ele deseja manter com ele e quanto ele deseja oferecer ao jogador $j$;

b. o jogador $j$ tem a opção de aceitar ou rejeitar a decisão;

c. se o jogador $j$ aceitar a decisão, cada pessoa recebe a sua parte na divisão proposta. Caso ele rejeite a oferta, nenhum dos dois recebe nada.

Esse jogo se tornou popular nos experimentos de economia, uma vez que ele é capaz de tratar comportamentos aparentemente irracionais. Guth et al. inicialmente supuseram que os dois jogadores seriam racionais, devido às consequências da negociação. Eles consideraram que mesmo se a proposta do jogador $i$ fosse "injusta", o jogador $j$ aceitaria, visto que "um pássaro na mão é melhor do que dois voando". Contudo, se o jogador $j$ concluir que o valor recebido é muito baixo, ele pode considerar isso um abuso e rejeitar a oferta. Assim, surgem algumas questões (GALE; BINMORE; SAMUELSON, 1995; THALER, 1988): qual é o valor mínimo que o jogador $j$ aceitaria? Qual é a melhor estratégia?

Existem muitos estudos (THALER, 1988; BOLTON; RAMI, 1995; CROSON; BUCHAN, 1999; GOSPIC et al., 2011; MARCHETTI et al., 2011; NOWAK; PAGE; SIGMUND, 2000; SANFEY et al., 2003; SILVA; KELLERMAN, 2007; SRINIVASAN et al., 2003; ZAK et al., 2009) que utilizam o jogo do ultimato como base teórica a fim de modelar as ações de acordo com os movimentos caracterizados nesse jogo.

### 4.3.2 Descrição das operações

O TrustUM estima a confiança entre qualquer par de nós mesmo se eles não tiveram nenhuma interação prévia entre si. Ele é dividido em três fases independentes mas complementares: monitoramento, troca de informações e avaliação de confiança. Durante as três fases, o TrustUM utiliza as duas estruturas seguintes:

a. **matriz de confiança**: um armazenamento para todas as informações obtidas. Também serve como banco de dados para as avaliações de confiança; e

b. **lista de confiança:** os dados obtidos pelos nós via observações diretas. É usada durante as trocas de informações.

A rede é modelada como um grafo $G(V, E_t)$, em que o conjunto dos vértices $V$ representa os nós e o conjunto das arestas $E_t$ representa as relações no tempo $t$. Todos os nós em $V$ são modelados como jogadores do jogo do ultimato. Cada jogador $i \in V$ tem uma Matriz de Confiança, que é considerada uma estratégia de grupo $S_i$. Uma estratégia

$s_i = (s_i^1, ..., s_i^n) \in S_i$, do jogador $i$, é um vetor de tamanho $n = |V|$, em que $s_i^j = 1$ se $i$ confia em $j$, $s_i^j = -1$ se $i$ não confia em $j$, e $s_i^j = 0$ se $i$ não possui informações sobre $j$. Cada nó tem uma Matriz de Confiança composta por $|N|$ vetores, para armazenar as informações de confiança de todos os nós.

#### 4.3.2.1 Monitoramento

O monitoramento é a primeira etapa do TrustUm. Ele consiste na avaliação dos vizinhos físicos. Existem muitas propostas na literatura que consideram a avaliação de confiança de nós vizinhos, tais como (CHO; SWAMI; CHEN, 2011; MEJIA et al., 2011; ZHENG; JIANG; BARAS, 2011; ZOURIDAKI et al., 2006). O TrustUm pode utilizar qualquer uma dessas abordagens. Na apresentação original, os autores utilizaram a abordagem do *watchdog* ("cão de guarda") proposta em (MEJIA et al., 2011).

#### 4.3.2.2 Troca de informações

Nessa fase os nós atualizam as suas Matrizes de Confiança via troca de dados com os seus vizinhos físicos. A troca de informações utiliza mensagens de *broadcast* limitadas a um salto de distância, sem a necessidade de protocolos de roteamento. Note que a convergência da rede depende da mobilidade dos nós. Além disso, os nós enviam apenas as informações que eles obtiverem via observações diretas na fase de monitoramento. Dessa forma, eles não retransmitem informações obtidas de outros nós.

Ao receber uma mensagem de troca de informação do nós $N_j$, o nó $N_i$ analisa o *Status* de Confiança da relação entre ele e o nó $N_j$. O *Status* de Confiança é baseado no Jogo do ultimato e é calculado com base na Matriz de Confiança. Ele determina a quantidade de informações que os nós $N_i$ e $N_j$ irão trocar.

A avaliação do *Status* de confiança da relação entre os nós $N_i$ e $N_j$ é baseada no Jogo do ultimato, e é utilizada como pagamento dos jogadores. O pagamento (ou *Status* de Confiança) do nó $N_i$ em relação ao nó $N_j$ é:

$$Payoff_i(j) = \frac{m_i}{m_i + m_j}, \tag{4.3}$$

em que $m_i$ e $m_j$ são definidos como:

$$m_x = \frac{\sum_{k=0}^{n} s_x^k}{\sum_{k=0}^{n} |s_x^k|} \tag{4.4}$$

em que $m_k$ é a soma das estratégias $s_k^x$ da Matriz de Confiança $S_x$ dividida pelo número de nós com $s_k^x \neq 0$. Em outras palavras, a quantidade de informação recebida por $k$ é proporcional ao $payoff_x(k)$, enquanto $x$ recebe as outras partes da proporção. Se o nó $N_i$ identifica o nó $N_j$ como não confiável, a informação será descartada.

O *Status* de Confiança indica os nós mais confiáveis. Quanto maior o valor do *Status* de Confiança, maior é a concentração de informações confiáveis que o nó armazena. Assim, ele receberá poucos dados dos nós com valores de *Status* de Confiança baixos.

Após a definição do *Status* de Confiança a troca é realizada e as informações recebidas são salvas na Matriz de Confiança. De forma similar à fase de monitoramento, a troca de informações acontece em intervalos de tempo $\Delta T_{ex}$.

#### 4.3.2.3 Avaliação de confiança

A avaliação de confiança pode ser realizada em qualquer momento. Contudo, o valor retornado pode variar no tempo, devido as trocas de informações. As variáveis que englobam a avaliação de confiança são:

a. **informação de $N_i$ sobre $N_j$ ($s_i^j$):** é a informação mantida na Matriz de Confiança e obtidas nas fases de monitoramento e trocas de informação. Baseado nessas informações, o valor de confiança será calculado;

b. **idade da informação ($\tau$):** determina a idade da informação armazenada na Matriz de Confiança;

c. **quantidade de nós ($n = |V|$):** representa o tamanho da estrutura e a quantidade de informação armazenada; e

d. **pesos ($p_1$ e $p_2$):** $p_1$ pode ser usando para aumentar/diminuir o efeito da idade da informação, enquanto $p_2$ é responsável por aumentar/diminuir a ênfase da informação obtida diretamente pelos nós.

A avaliação da confiança de $N_i$ e $N_j$ no tempo $t$ é dada por:

$$Trust_i(j) = \frac{\sum_{k=0}^{n}(s_k^j \cdot (1 - \frac{t-\tau_k^j}{t}) \cdot p_1)}{\sum_{k=0}^{n} |s_k^j|} + s_i^j \cdot p_2 \tag{4.5}$$

em que $1 - \frac{t-\tau_k^j}{t}$ corresponde à relação entre o tempo da troca da informação $\tau_k^j$ e o tempo atual $t$. Em outras palavras, quanto mais recentes são as informações maior será o valor resultante, variando no intervalo de $[-1, 1]$. Note que $k$ deve ser necessariamente diferente de $N_i$. A idade não é aplicável para as informações obtidas na fase de monitoramento, já que o nó $N_i$ é totalmente responsável pela legitimidade dessa informação.

Além disso, é possível utilizar diferentes pesos $p_1$ e $p_2$ na avaliação da confiança. Alterando $p_1$, é possível aumentar/diminuir os efeitos da idade da informação no resultado final. Por outro lado, alterar $p_2$ afeta o peso das informações obtidas na fase de monitoramento.

## 4.4 Conclusão

Este capítulo apresentou duas propostas que podem ser utilizadas como ferramenta para o gerenciamento confiança no SEMAN. Tanto o TRUE como o TrustUM são duas abordagens que visam fornecer uma boa técnica para a avaliação de confiança dos nós considerando o contexto das aplicações.

Os demais componentes do *middleware*, como o gerenciamento de chaves, por exemplo, podem utilizar os serviços do gerenciamento de confiança para construir relações confiáveis. Dessa forma, as duas abordagens apresentadas visam a oferecer uma forma para que os demais módulos do SEMAN possam garantir maior segurança aos serviços oferecidos.

# CAPÍTULO 5

# GERENCIAMENTO DE CHAVES

A principal dificuldade em adotar IBC em MANETs é que uma entidade centralizada é necessária para atuar como um PKG, que viola a natureza auto-organizada dessas redes. Além disso, um PKG centralizado requer uma entidade confiável, e ele pode ser um ponto único de falha no sistema. Em uma PKI tradicional, baseado em certificados, se a *Certificate Authority* (CA) gera um certificado falso para um cliente contendo uma chave pública falsa, o cliente é capaz de detectar e provar o comportamento malicioso do CA. Também, o CA não pode decifrar uma mensagem cifrada pelo cliente e ele não pode assinar qualquer mensagem em nome do cliente, caso o receptor tenha obtido um certificado correto do cliente.

Além disso, em um IBC, como o PKG conhece a chave privada mestre, ele é capaz de decifrar ou assinar mensagens em nome de qualquer cliente, sem qualquer ataque ativo e sem ser detectado. Esse problema é conhecido como "custódia de chaves" [1]. Assim, os serviços do PKG em um IBC deve ser mais confiável do que uma conhecida CA em um PKI tradicional baseado em certificados. Essa questão tem sido considerada como a razão pela baixa adoção do IBC fora de ambientes organizacionais fechados (KATE; GOLDBERG, 2010). Boneh e Franklin sugeriram distribuir o PKG para gerenciar esses problemas usando esquemas de compartilhamento de segredo $(n, t)$, em que $n$ nós formam um *Distributed PKG* (D-PKG) e apenas um subconjunto de $t+1$ nós é capaz de computar a chave privada mestre (BONEH; FRANKLIN, 2001).

Diversos esquemas de gerenciamento de chaves baseados em identidade têm sido projetados para as MANETs e a maioria deles consideram a distribuição do PKG. Contudo, os esquemas propostos não consideram todas as características dessas redes. A próxima seção apresenta um estudo das principais soluções de gerenciamento de chaves baseada em

[1] Termo comumente encontrado na literatura como *key escrow*.

identidade para MANETs e as demais seções descrevem o novo esquema de gerenciamento de chaves proposto neste trabalho.

## 5.1 Trabalhos relacionados

Diversos esquemas de gerenciamento de chaves baseados em identidade podem ser encontrados na literatura. Essa seção apresenta os esquemas mais importantes para as MANETs, discutindo as suas abordagens, forças e fraquezas.

O esquema de Khalili-Katz-Arbaugh (KHALILI; KATZ; ARBAUGH, 2003) combina as técnicas de criptografia baseada em identidade e criptografia de limiar. Todos os $n$ nós que inicializam a MANET formam um PKG distribuído, cuja chave privada mestre é distribuída entre eles usando um esquema compartilhamento do segredo $t$-sobre-$n$. A chave pública mestre é disponibilizada publicamente na rede. Para receber a sua chave privada, um nós apresenta sua identidade para no mínimo $t$ nós do D-PKG e cada um deles envia uma parte da chave privada novamente ao nó requisitante. Quando esse nó recebe $t$ partes corretas, ele constrói a sua chave privada.

O esquema de Khalili-Katz-Arbaugh assume que as identidades são armazenadas em um hardware resistente a alteração. Um atacante que criar uma identidade falsa ou alterar sua própria identidade pode ser uma ameaça para o esquema. Os nós que entram na rede precisam de um canal de comunicação seguro com no mínimo $t$ nós do PKG distribuído para obter suas respectivas chaves privadas. Além disso, esse esquema não considera a revogação e a renovação de chaves.

O esquema de Deng-Mukherjee-Agrawal (DENG; MUKHERJEE; AGRAWAL, 2004) tem dois componentes: uma geração distribuída de chave e uma autenticação baseada em identidade. A geração de chave fornece a chave mestre da rede os pares de chaves pública e privada de cada nó da rede. A autenticação baseada em identidade fornece autenticação fim-a-fim e confidencialidade entre os nós. Se o processo de autenticação for bem sucedido, esses nós trocam uma chave de sessão, que pode ser usada em comunicações futuras. As chaves privada/pública mestre são computadas e distribuídas da mesma forma que o esquema de Khalili-Katz-Arbaugh.

Para garantir que as partes geradas da chave privada são transmitidas de forma segura, o nó requisitante deve gerar e apresentar uma chave pública temporária quando está enviando o pedido. Cada nó do D-PKG envia a subparte da chave privada cifrada com essa chave pública temporária. Contudo, como no esquema de Khalili-Katz-Arbaugh, um atacante que cria identidades falsas ou altera a sua própria identidade pode ser uma ameaça. Além disso, esse esquema também não considera a revogação e a renovação de chaves.

O esquema de Bohio-Miri (BOHIO; MIRI, 2004) usa chaves simétricas computadas não interativamente pelos nós. Ele assume que todos os nós estão pré-configurados corretamente, com os parâmetros públicos do sistema e com suas respectivas chaves privadas, antes da formação da rede. Quando dois nós desejam se comunicar, eles calculam uma chave simétrica compartilhada, usando uma função *hash*. Esse processo é chamado de *a*cordo de chaves. O acordo de chaves é não interativo e não requer o envolvimento do PKG. Para reduzir a sobrecarga de comunicação imposta pelo acordo de chaves, os autores sugerem o uso de chaves simétricas de *broadcast*.

Como as chaves de nós e de *broadcast* são simétricas, o esquema não possui irretratabilidade e permite ataques de personificação. Para garantir a irretratabilidade e tratar ataques de personificação, um esquema de assinatura baseada na difusão de um segredo foi proposto. Contudo, esse esquema de assinatura é vulnerável a ataques de falsificação (CHIEN; LIN, 2008). Esse esquema também não considera a revogação e a renovação de chaves. Além disso, ele viola o espírito da criptografia baseada em identidade, quando requer uma estrutura de suporte e servidores *online* (CHIEN; LIN, 2008).

Um outro esquema, o *identity-based authentication and key exchange* (IDAKE) (HOEPER; GONG, 2006a) consiste de duas abordagens: IDAKE básico e IDAKE totalmente auto-organizado. Ele usa criptografia simétrica e chaves baseadas em emparelhamento nas duas abordagens, que são especificadas em seis algoritmos: configuração, extração, distribuição, cálculo da chave compartilhada, atualização de chave e revogação de chave.

O IDAKE básico consiste de duas fases: a inicialização que acessa um PKG externo (algoritmos de configuração, extração e distribuição) e a execução do sistema sem acesso

ao PKG (algoritmos de cálculo das chaves compartilhadas, renovação de chave e revogação de chave). Note que o PKG externo deve inicializar todos os dispositivos antes que eles acessem a rede. No IDAKE totalmente auto-organizado, todas as tarefas são executadas pelos próprios nós, sem um PKG externo. O PKG externo é emulado por um esquema de limiar $t$-sobre-$n$, como os esquemas já apresentados. Contudo, a versão auto-organizada não especifica como as chaves privadas são distribuídas para os nós.

O *Identity-based Key Management* (IKM) (LIU, 2006) é uma combinação do criptografia de limiar com o gerenciamento de chaves baseado em identidade. No IKM, as chaves pública e privada de cada nó são compostas por um elemento baseado em identidade específico do nó e um elemento comum de toda a rede. O elemento específico do nó garante que o sigilo dos nós não comprometidos não é prejudicado mesmo na presença de diversos nós comprometidos. Por outro lado, o elemento comum de toda a rede permite uma eficiente atualização de chaves usando uma única mensagem de *broadcast*.

O IKM possui três fases: pré-distribuição de chave, revogação de chave e atualização de chave. A pré-distribuição de chave ocorre durante a inicialização da rede, em que o PKG determina um conjunto de parâmetros do sistema e pré-carrega todos os nós com o material criptográfico apropriado. O PKG distribui sua funcionalidade para $t$ nós que formam o PKG distribuído. A chave privada mestre é distribuída usando um esquema de criptografia de limiar $t$-sobre-$n$. Isso é feito para permitir a revogação e atualização segura e robusta de chaves durante as operações da rede.

As revogações de chaves podem ser explícitas a fim de minimizar o dano das chaves comprometidas, Durante a operação da rede, se qualquer nó suspeitar que um outro nó é malicioso ou foi comprometido, ele envia uma mensagem assinada ao PKG distribuído. Um nó é considerado malicioso quando o número de acusações contra ele alcança um valor pré-definido, chamado de limiar de revogação. No IKM, os nós devem atualizar suas chaves pública/privada em intervalos periódicos ou quando o número de nós revogados alcança um valor pré-determinado. Os nós revogados não podem atualizar as suas chaves, sendo isolados da rede. Uma avaliação do IKM foi realizada para a formulação do novo esquema de gerenciamento de chaves e publicada em (SILVA; LIMA; ALBINI, 2010) e

(SILVA; ALBINI; LIMA, 2013).

## 5.2 O esquema *i*FUSO

Essa seção apresenta o *Identity-Based Fully Self-Organized Key Management for MANETs* (*i*FUSO), o esquema de gerenciamento de chaves proposto para ser integrado ao SEMAN, e publicado em (SILVA; ALBINI, 2013). O *i*FUSO considera uma rede assíncrona composta por $n$ nós, representados por $N_1, N_2, ..., N_n$, sendo que os nós maliciosos podem comprometer no máximo $t$ nós, sendo $t < n$. Além disso, é considerado que apenas nós confiáveis participam da inicialização do sistema. Os nós que inicializam o sistema são chamados de nós fundadores, denotados por $N_{\mathcal{F}}$. Esses nós fundadores formam, auto-organizadamente, o D-PKG. Nenhum nó na rede conhece a chave mestre do sistema, uma vez que ela é distribuída em um esquema de limiar $t$-sobre-$n$. Também, para se adaptar ao dinamismo da rede, ele permite que os nós entrem e saiam do D-PKG.

Para prevenir o sistema de ataques de criptanálise, o *i*FUSO fornece uma forma de atualizar as chaves públicas e privadas dos nós, similar a (LIU, 2006; LUO et al., 2004). A atualização de chave acontece periodicamente de acordo com um intervalo pré-determinado, ou reativamente quando o número de nós revogados alcança um valor de limite. Os nós são capazes de atualizar suas chaves públicas autonomamente e suas chaves privadas requisitando ao D-PKG. Além disso, o *i*FUSO suporta revogações implícitas e explícitas. As revogações implícitas são baseadas no tempo de expiração das chaves privadas, enquanto as revogações explícitas são baseadas em declarações emitidas pelo D-PKG após uma quantidade pré-definida de acusações contra um nó malcomportado.

A Tabela 5.1 resume a notação considerada para descrever o esquema de gerenciamento de chaves proposto nesta tese.

### 5.2.1 Inicialização

O *i*FUSO deve ser inicializado por um conjunto de nós fundadores ($N_{\mathcal{F}}$), $N_{\mathcal{F}} = m$. *i*FUSO possui apenas uma suposição: os nós fundadores devem ser capazes de trocar infor-

Tabela 5.1: Notação do gerenciamento de chaves

| Notação | Descrição |
|---|---|
| $\mathbb{G}_1$ | grupo aditivo cíclico de ordem prima $p$ |
| $\mathbb{G}_2$ | grupo multiplicativo cíclico de ordem prima $p$ |
| $e$ | um emparelhamento bilinear em que $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ |
| $H_1(x)$ | função *hash* em que $H_1(x) = \{0,1\}^* \rightarrow \mathbb{G}_1^*$ |
| $H_2(x)$ | função *hash* em que $H_2(x) = \mathbb{G}_2 \rightarrow \mathbb{Z}_p^*$ |
| $N_{\mathcal{F}}$ | nós fundadores |
| $N_i$ | identificação do nó $i$ |
| $SK_i$ | chave privada do nó $i$ |
| $PK_i$ | chave pública do nó $i$ |
| $MSK$ | chave privada mestre do sistema |
| $MPK$ | chave pública mestre do sistema |
| $MSK_i$ | arte da chave privada mestre mantida pelo nó $i$ |

mações de forma segura para inicializar o sistema. Como primeiro passo da inicialização, os nós devem determinar:

a. o tamanho do sistema $m$ e o limiar de segurança $t$;

b. $p$ e $q$: dois números primos grandes, em que $q$ divide $(p-1)$;

c. $\mathbb{G}_1$: um grupo aditivo cíclico de ordem prima $p$;

d. um gerador $g \in \mathbb{G}_1$;

e. $\mathbb{G}_2$: um grupo multiplicativo cíclico de ordem prima $p$;

f. o tipo de emparelhamento a ser usando e selecionar dois grupos $\mathbb{G}_1$, e $\mathbb{G}_2$, tal que exista um emparelhamento bilinear $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ do tipo escolhido; e

g. $\mathcal{G} = \langle e, \mathbb{G}_1, \mathbb{G}_2 \rangle$.

Esse passo pode ser executado juntamente pelos nós que inicializam o sistema ou proposto por um nó aos demais.

Após essa etapa, cada $N_i \in N_{\mathcal{F}}$ deve ter os seguintes elementos públicos:

a. os números primos $p$ e $q$;

b. o gerador $g$ e o grupo aditivo cíclico $\mathbb{G}_1$;

c. $\mathbb{Z}_q^*$: um função elíptica de ordem prima $q$;

d. $H_1(x)$: uma função *hash* em que $H_1(x) = \{0,1\}^* \rightarrow \mathbb{G}_1^*$; e

e. $H_2(x)$: uma função *hash* em que $H_2(x) = \mathbb{G}_2 \rightarrow \mathbb{Z}_p^*$.

Para inicializar o sistema, os nós fundadores devem configurar o D-PKG, que consiste na geração da chave pública mestre e sua chave privada mestre correspondente. O D-PKG é construído em um esquema distribuído $t$-sobre-$m$ entre os $m$ nós fundadores. A configuração do D-PKG no $i$FUSO é composta pelos seguinte passos:

a. cada $N_i \in N_{\mathcal{F}}$ escolhe aleatoriamente uma função polinomial simétrica de duas variáveis $f_i(x,y)$ sobre $\mathbb{Z}_q$ em que as duas variáveis $x$ e $y$ devem ser de ordem máxima $t$. A função polinomial pode ser descrita como:

$$f_i(x,y) = \sum_{k=0}^{t} \sum_{j=0}^{t} a_{k,j}^i x^k y^j, \tag{5.1}$$

em que $a_{k,j}^i \in \mathbb{Z}_q$, $a_{k,j}^i = a_{j,k}^i$ e $a_{0,0}^i = z_i$;

b. cada $N_i \in N_{\mathcal{F}}$ calcula $f_l^i(x) = f_i(x,l)$ para todo $N_l \in N_{\mathcal{F}}$ as:

$$f_l^i(x) = f_i(x,l) = \sum_{k=0}^{t} \sum_{j=0}^{t} a_{k,j}^i x^k l^j, \tag{5.2}$$

então, $N_i$ envia seguramente $f_l^i(x)$ to $N_l$;

c. cada $N_i$ calcula sua parte da chave privada $MSK_i$:

$$MSK_i = f_i(x) = \sum_{j=1}^{n} f_j^i(x) = \sum_{j=1}^{n} f_j(x,i) = f(x,i) \tag{5.3}$$

d. a chave privada mestre $MSK$ não é calculada nem conhecida por nenhum nó, mas é igual a:

$$MSK = \sum_{N_i \in N_{\mathcal{F}}} MSK_i \bmod q \tag{5.4}$$

Cada nó $N_i$, após calcular sua parte $MSK_i$, publica $g^{MSK_i}$, sendo que $g$ é um parâmetro comum usado pela criptografia baseada em identidade (BONEH; FRANKLIN, 2001). Uma vez que um nó recebe $t$ partes, a chave pública mestre pode ser calculada como $MPK = \sum_{i=1}^{t} g^{MSK_i}$. Note que $MPK$ pode ser disponibilizada publicamente para todos os nós da rede.

### 5.2.2 Associação de novos membros ao D-PKG

Em uma MANET é muito importante que um D-PKG seja altamente dinâmico e descentralizado, e que os novos nós sejam capazes de participar do PKG distribuído em qualquer momento. Para isso, esses novos nós devem receber uma parte da chave privada mestre $MSK$, calculada por, no mínimo, $t$ membros do D-PKG.

Se um novo nó $N_k$ deseja participar do D-PKG, ele deve contactar no mínimo $t$ membros desse D-PKG para obter as informações necessárias desses nós. A associação de novos nós ao D-PKG deve ser realizada da seguinte forma:

a. nó $N_k$ seleciona $t$ membros do D-PKG, representado por $\Omega$;

b. nó $N_k$ envia um pedido de aceitação como membro do D-PKG a cada nó de $\Omega$;

c. cada nó $N_j \in \Omega$ envia uma parte da informação $f(j,k) = S_j^k$ para $N_k$; e

d. após receber $t$ respostas, $N_k$ pode calcular sua parte polinomial $MSK_k$ usando uma interpolação de Lagrange:

$$MSK_k = S_k(x) = \sum_{j=1}^{t} \lambda_j S_j^k = \sum_{j=1}^{t} \lambda_j f(j,k) = f(x,k) \tag{5.5}$$

Na etapa *c*, o nó $N_j$, membro do D-PKG somente envia a parte da informação $f_{j,k}$, se ele considerar o nó $N_k$ confiável e apto para participar das operações do *i*FUSO. Após construir $MSK_k$, $N_k$ possui as informações necessárias para participar de todas as operações de gerenciamento de chaves.

### 5.2.3 Emissão de chave privada dos nós

O *i*FUSO é composto por um número contínuo de fases não sobrepostas de atualização de chaves, denotadas por $p_\Delta$, em que $\Delta$ representa o índice da fase. Como em (LIU, 2006), cada $p_\Delta$ é associado com uma cadeia binária única, denotada como $str_\Delta$. Para cada nó $N_i$, sua chave pública é representada por $PK_i = H_1(N_i)$ enquanto a chave privada correspondente por $SK_i = (PK_i)^{MSK}$, sendo que $MSK$ é a chave privada mestre. Relembrando, no *i*FUSO nenhum nó conhece a $MSK$, que é gerada e armazenada de forma totalmente distribuída. Para obter a sua chave privada $SK_i$, o nó $N_i$ deve solicitá-la ao D-PKG e esperar no mínimo $t$ respostas corretas. Assim, os seguintes passos devem ser realizados:

a. $N_i$ seleciona no mínimo $t$ nós do D-PKG. Esse conjunto de nós é denotado por $\Psi$. Para minimizar o tempo de requisição, $\Psi$ pode conter todos os nós do D-PKG;

b. $N_i$ solicita sua parte de $SK_i$ para cada nó de $\Psi$;

c. cada nó $N_j \in \Psi$ envia uma parte da chave privada $\sigma_i^j = (PK_i)^{MSK_j}$ para o nó$N_i$; e

d. ao receber $t$ respostas corretas, $N_i$ pode calcular a sua chave privada $SK_i$ como:

$$SK_i = \prod_{k \in \Psi} (\sigma_i^k)^{\lambda_k}, \tag{5.6}$$

em que $\lambda_k = \prod_{k \in \Psi} \frac{k}{k-i}$ são coeficientes de Lagrange apropriados (KATE; GOLDBERG, 2010).

Note que se $N_i$ é um membro do D-PKG, ele precisa receber apenas $t-1$ respostas para construir a sua chave privada, já que ele pode calcular a sua parte $\sigma_i^i = (PK_i)^{MSK_i}$.

### 5.2.4 Atualização de chaves

Para prevenir ataques contra o D-PKG e ameaças resultantes das chaves comprometidas, uma técnica similar à proposta em (LIU, 2006), conhecida como atualização de

chaves, é empregada no *i*FUSO. As soluções de segurança baseadas em atualização de chaves são comuns nas MANETs (ZHOU; HAAS, 1999; KONG et al., 2001; YI; KRAVETS, 2003). No *i*FUSO, uma nova fase de atualização de chaves $p_{i+1}$ inicia após um tempo pré-determinado. Como todos os nós devem atualizar as suas chaves, se os membros do D-PKG não atualizam a chave de um dado nó $N_a$, ele é considerado [implicitamente] revogado.

As chaves pública e privada de um nó $N_a$ são representadas por tuplas, respectivamente, $\langle PK_a, H_1(str_i)\rangle$ e $\langle SK_a, MSK_{p_i}\rangle$. $H_1(str_i)$ representa uma informação pública do D-PKG associada com a fase $p_i$ e $SK_{p_i}$ representa uma informação de fase comum gerada pelos membros do D-PKG. Por simplicidade, a chave pública do nó $N_a$ durante a fase $p_i$ é denotada como $PK_{a,p_i} = \langle H_1(N_a), (H_1(str_i)\rangle$ e a sua chave privada correspondente é $SK_{a,p_i} = \langle PK_a^{MSK}, H_1(str_i)^{MSK}\rangle = \langle SK_a, MSK_{p_i}\rangle$.

Cada nó $N_a$ pode, autonomamente, atualizar sua chave pública $PK_{a,p_i} = (H_1(N_a), H_1(str_i))$, em que $str_i = str_{i-1} + 1$. Por outro lado, gerar a chave privada da fase envolve no mínimo $t$ membros do D-PKG. Um dado nó $N_z$, membro do D-PKG, envia um pedido para $t-1$ outros membros do D-PKG. Sendo $\Phi$ esse conjunto de nós selecionados, includindo o nó $N_z$. Então, cada $N_i \in \Phi$ gera um elemento de chave privada comum parcial $H_1(str_i)^{MSK_i}$, e a envia para o nó $N_z$. Após receber $t$ elementos parciais, $N_z$ constrói a $H_1(str_i^{MSK})$ completa usando uma interpolação de Lagrange:

$$MSK_{p_i} = \sum_{i \in \Phi} \lambda_i(0) H_1(str_i)^{MSK_i} = H_1(str_i)^{MSK} \tag{5.7}$$

Após isso, o elemento de chave privada comum deve ser enviado para todos os nós do sistema. Para impedir que os nós revogados participem das operações criptográficas, o elemento de chave privada atualizado não deve ser disponibilizado para esses nós. Assim, é utilizada uma variante do protocolo de difusão cifrada[2] baseada em identidade proposta em (HUR; PARK; HWANG, 2012).

Sendo $\mathcal{R}$ o conjunto de nós revogados, então o nó $N_z$ gera os parâmetros da difusão cifrada:

[2] *broadcast encryption*

1) $\forall i \in \mathcal{N} \setminus \mathcal{R}$ calcula $PK_i = H_1(N_i)$;

2) seleciona aleatoriamente $r \in \mathbb{Z}_p^*$ e $\forall i \in \mathcal{N} \setminus \mathcal{R}$ calcula $s_i = H_2(\hat{e}(PK_i^r, MPK))$;

3) seleciona aleatoriamente $k \in \mathbb{Z}_p^*$ e calcula uma chave de cifração de mensagem $K = \hat{e}(g,g)^k$;

4) seleciona aleatoriamente $\alpha \in \mathbb{Z}_p^*$; e

5) calcula $Hdr = (C_1, C_2, C_3)$ em que

$$C_1 = g^r; C_2 = (g^\alpha)^k; C_3 = \{c_i = (g^{1-\frac{1}{\alpha}})^{\frac{1}{s_i}}\}_{N_i \in \mathcal{N} \setminus \mathcal{R}}$$

Então, $N_z$ tem a chave $K$ e $Hdr$, e usa $K$ para cifrar o elemente de chave privada comum $MSK_{p_i}$, gerando $C_K$. Finalmente, $N_z$ difunde na rede a mensagem cifrada $CM = (Hdr, C_K)$.

Quando um nó não revogado $N_b$ recebe essa mensagem, ele é capaz de obter a chave de cifração de mensagem $K$ encapsulada no cabeçalho $Hdr$, usando sua chave privada correspondente $SK_b$, como segue:

1) calcula $s_i = H_2(\hat{e}(SK_b, C_1))$; e

2) obtém $c_i$ de $C_3$ e calcula

$$\hat{e}(C_2^{-1}, c_i^{s_i}) \times \hat{e}(g, C_2)$$

$$= \hat{e}(((g^\alpha)^k)^{-1}, ((g^{1-\frac{1}{\alpha}})^{\frac{1}{s_i}})^{s_i}) \times \hat{e}(g, (g^\alpha)^k) =$$

$$= \hat{e}(g,g)^{-k(\alpha-1)} \times \hat{e}(g,g)^{k\alpha} = K$$

Com $K$, o nó $N_b$ pode decifrar a mensagem cifrada $C_K$, extraindo $MSK_{p_i}$.

### 5.2.5 Revogação de chaves

O *i*FUSO também fornece técnicas para verificar se a chave pública de um dado nó está revogada. As revogações de chaves públicas devem ser mantidas dentro do sistema,

já que os nós devem ser capazes de verificar imediatamente o *status* de uma chave pública (HOEPER; GONG, 2006b). A maioria dos esquemas de gerenciamento de chaves para MANETs considera a revogação de chave baseada em tempo de expiração (DAZA; MORILLO; RàFOLS, 2007). Contudo, essa abordagem não é suficiente pois os nós devem ser capazes de revogar chaves antes que elas expirem, como consequência de um comprometimento de chave ou comportamento malicioso.

Assim, o *i*FUSO suporta tanto a revogação implícita quanto a explícita. Se um nó não pode obter o elemento de chave privada comum durante uma dada fase $p_i$, então ele não será capaz de cifrar ou decifrar qualquer informação durante essa fase, e é considerado implicitamente revogado.

Por outro lado, a revogação explícita do *i*FUSO é baseada em uma lista de nós revogados armazenada pelos próprios nós. Quando um nó $N_b$ detecta o mau comportamento de um nó $N_a$, ele gera uma mensagem de acusação assinada contra $N_a$, que deve ser enviada aos nós do D-PKG. Para evitar a interceptação da mensagem de acusação, ela é enviada via difusão cifrada para os membros do D-PKG, como detalhado na seção 5.2.4. Essa técnica, além de diminuir o custo de comunicação da revogação, aumenta a segurança, já que os nós maliciosos não serão capazes de ler a mensagem de revogação.

Ao receber uma mensagem de acusação do nó $N_b$ contra $N_a$, um membro do D-PKG descarta essa mensagem caso o próprio nó $N_b$ já tenha sido previamente revogado. Caso contrário, ele salva a mensagem de acusação. Para prevenir falsas acusações contra nós legítimos, um nó $N_a$ é diagnosticado como comprometido apenas quando as acusações contra ele alcancem um limite $\gamma$ em um intervalo de tempo pré-determinado. O valor de $\gamma$ define o *trade-off* entre a tolerância a falsas acusações e a detecção de nós comprometidos.

Quando o limite de revogação é alcançado, uma revogação de chave contra o nó $N_a$ é gerada e publicada. Supondo que $\Phi \in \Omega$ é o conjunto de pelo menos $t$ membros do D-PKG que receberam $\gamma$ acusações contra $N_a$. Então, cada nó $N_v \in \Phi$ emite uma mensagem de revogação parcial $MSK_v H_1(N_a)$ e a envia para os demais membros do D-PKG. Esses nós

podem construir a revogação completa usando uma interpolação de Lagrange, como:

$$\overline{N_a} = \sum_{V \in \Phi} \lambda_V(0) MSK_v H_1(N_a) = MSKH_1(N_a) \bmod q$$

em que $\lambda_V(0)$ são os coeficientes de Lagrange apropriados.

Após construir a revogação de chave contra $N_a$, o membro de $\Phi$ com o menor $ID$ difunde $\langle N_a, \overline{N_a} \rangle$ na rede, para informar que $N_a$ está comprometida e sua chave foi revogada. Ao receber uma revogação contra o nó $N_a$, todos os nós verificam a sua validade e a armazenam localmente.

## 5.3 Prova de segurança

Essa seção descreve as provas de segurança das principais operações fornecidas pelo *i*FUSO. Esta segurança está relacionada a como a inicialização do sistema e a associação de novos membros garante a exatidão e o sigilo para os usuários. Por outro lado, está fora do escopo dessa avaliação mostrar que as operações de emissão de chaves privada, cifração e decifração são criptograficamente seguras, uma vez que elas são baseadas na solução de Boneh e Franklin (BONEH; FRANKLIN, 2001) e, portanto, herdam essas características de segurança.

### 5.3.1 Inicialização

A segurança da inicialização do sistema é definida em termos de exatidão. A propriedade da exatidão requer que, mesmo que existam no máximo $d \leq t$ nós desonestos, os nós fundadores obtêm corretamente sua parte da chave privada $MSK$.

**Lemma 1.** *Todo o subconjunto de t partes fornecidos pelos nós geram a mesma única chave privada mestre MSK.*

*Demonstração.* Prova similar à apresentada em (QIAN et al., 2011)

Suponha que cada $N_i \in N_F$ tenha realizado com sucesso o compartilhamento de $z_i$ entre todos os nós, e cada nó $N_j \in N_F$ receba sua parte $f_i^j = f_i^j(0) = f_j(0, i)$ sobre $z_i$.

Com essas partes, obtém-se um polinômio único $f_i(0,x)$ que satisfaz $f_i(0,0) = z_i$.

Assim, para qualquer conjunto $R$ de $t$ partes corretas, $z_i = \sum_{j \in R} \gamma_j f_i^j$ em que $\gamma_i$ são os coeficientes da interpolação de Lagrange apropriados para o conjunto $R$. Já que cada nó $N_j$ calcula sua parte $MKS_j = \sum_{i \in N_F} f_i^j$, tem-se que para o conjunto $R$:

$$MSK = \sum_{j \in R} z_j = \sum_{j \in N_F} \sum_{k \in R} \gamma_k s_{jk} = \sum_{k \in R} \gamma_k \sum_{j \in N_F} f_j^k = \sum_{k \in R} \gamma_k MSK_k$$

Sendo que isso vale para todo conjunto de $t$ partes corretas, então $MSK$ é definida unicamente. □

**Lemma 2.** *$MSK$ é uniformemente distribuída em $\mathbb{Z}_q$*

*Demonstração.* O segredo $MSK$ é definido como $MSK = \sum_{i \in N_F} z_i$. Note que como o valor de $z_i$ nesse somatório é escolhido de forma aleatória e independente entre os demais nós do somatório, é necessário garantir que exista uma distribuição uniforme de $MSK$.

Sendo $N_{i \in N_F}$ um nó honesto e supondo que um adversário controle $t$ partes de $f_i(x,y)$. Sem a perda da generalidade, assume-se que o adversário conhece $f_i^1(x), f_i^2(x), ..., f_i^t(x)$ formando $View_A = \{f_i^1(x), f_i^2(x), ..., f_i^t(x)\}$. É trivial mostrar que, para qualquer valor $s$, pode-se encontrar $b_{ij} \in \mathbb{Z}_q$, em que $b_{00} = s, b_{ij} = b_{ji}, 0 \leq i,j \leq t$ tal que se $f_i(x,y) = \sum_{i=0}^{t} \sum_{j=0}^{t} b_{ij} x^i y^j$, então $f_i(x,1) = f_i^1(x), f_i(x,2) = f_i^2(x), ...., f_i(x,t) = f_i^t(x)$. Isto implica em que $View_A$ não contém qualquer informação de $z_i$.

Prob[$N_i$ ter o segredo $z_i | View_A$] = Prob[$N_i$ ter o segredo $z_i$] = $\frac{1}{q}$ para todo $z_i \in \mathbb{Z}_q$. Assim, é independente da visão do adversário que cada nó $N_i \in N_F$ escolhe $z_q$. Daí, a chave privada mestre $MSK$ é uniformemente distribuída. □

Os lemas 1 e 2 mostram a exatidão da inicialização do $i$FUSO.

### 5.3.2 Associação de novos membros ao D-PKG

A segurança da operação de associação de novos membros ao D-PKG é definida em termos de exatidão e sigilo. A propriedade da exatidão requer que, mesmo diante de no máximo $k$ ($k \leq t$) nós desonestos, os novos membros obtenham corretamente uma parte

da chave privada mestre $MSK$. Por sigilo, entende-se que um novo membro não obtém qualquer informação completa da chave privada mestre $MSK$; E, quaisquer $d \leq t$ nós não obtém a sub-parte da chave privada mestre $MSK$ que está sendo obtida pelo novo membro.

**Lemma 3.** *Um novo membro não obtém qualquer informação completa da chave privada mestre.*

*Demonstração.* Prova similar à apresentada em (QIAN; JIA, 2012)

Supondo que os nós $N_1, ..., N_n$ gerem, em conjunto, um polinômio simétrico de duas variáveis sobre $\mathbb{Z}_q$, como segue:

$$f(x, z) = \sum_{i=0,j=0}^{t} a_{ij} x^i z^j \tag{5.8}$$

em que $f(0, 0) = MSK$ e $a_{ij} = a_{ji}$. Assim, $f(x, z)$ é denotado como $f(x, z) = X^T AY$, em que $X = (1, x, ..., x^t)^T$ e $Y = (1, y, ..., y^t)^T$. $A$ é uma matriz simétrica e $X^T$ denota a transposta de $X$.

Um novo membro $N_e$ obtém uma parte $MSK_e$ da chave privada mestre $MSK$ e um polinômio $f_e(x) = \sum_{i=0}^{t} a_i x^i = X^T Z$ sobre $\mathbb{Z}_q$, sendo $Z = (a_0, a_1, ..., a_t)^T$.

Para provar que $N_e$ não obtém qualquer informação completa da chave privada mestre $MSK$, mostra-se que, dado qualquer $a \in \mathbb{Z}_q$, existe uma matriz simétrica $A_a$ e um polinômio simétrico de duas variáveis $f_a(x, y) = X^T A_a Y$ sobre $\mathbb{Z}_q$ que satisfaz $f_a(0, 0) = a$ e $f_a(e, x) \equiv f_m(x) \bmod q$.

Assumindo que $N_e$ é o novo membro, então $X^T Z \equiv X^T A_a E \bmod q$, em que $E = (1, e, ..., e^t)^T$ para qualquer $X \in \{1\} \times (\mathbb{Z}_q)^t$, $X^T(A_a E - Z \equiv 0 \bmod q$.

Assim $A_a E \equiv Z \bmod q$ que é um conjunto de entradas em relação a equação $A_a = (a_{ij})_{1 \leq i \leq t+1, 1 \leq j \leq t+1}$ como variáveis. Como $A_a$ é uma matriz simétrica e $a_{11} = a$, o conjunto das equações anteriores têm $\frac{(t+1)^2-(t+1)}{2} - 1$ variáveis livres. Assim, existe $q^{\frac{(t+1)^2-(t+1)}{2}-1}$ matrizes simétricas que satisfazem o conjunto de equações anteriores. □

**Lemma 4.** *Quaisquer $d \leq t$ nós não obtém uma informação de uma sub-parte de $MSK$*

*Demonstração.* É suficiente provar o lema para o caso em que $d = t$. Se $t$ nós não são capazes de obter uma sub-parte de $MSK$ que é de um novo membro $N_e$, então um número menor de nós também não será capaz. Assim, sem a perda da generalidade, sendo esses $t$ nós $N_1, ..., N_t$. Então cada $N_i$, $1 \leq i \leq t$, obtém $f_e(i) = f_i(e)$. Porque, para qualquer $a \in \mathbb{Z}_q$ pode-se construir um polinômio $f^0(x)$ que satisfaça $h^0(0) = a, h^0(1) = h_m(1), ..., h^0(t) = h_m(t)$. Assim, esses $t$ nós não podem obter $MSK_e$. □

Portanto, os lemas 3 e 4 juntos mostram que essa operação possui as propriedades de exatidão e sigilo.

## 5.4 Sobrecarga de comunicação

Essa seção demonstra a sobrecarga de comunicação de todas as operações. Todos os custos comunicação são medidos considerando o número trocas de mensagens entre os nós.

### 5.4.1 Inicialização

O custo de comunicação para inicializar o esquema de gerenciamento de chaves é diretamente proporcional ao número de nós no D-PKG. Como mencionado na seção 5.2.1, a fim de criar um D-PKG, cada membro gera uma função polinomial, calcula uma sub-parte do $MSK$ e a envia para os demais membros do D-PKG. Considerando um D-PKG com $m$ nós, o custo para inicializar o gerenciamento de chaves, denotado por $IC$, é:

$$IC = m.(m-1) \ . \ sizeof(f_m^i(x)) \tag{5.9}$$

em que $sizeof(f_m^i(x))$ é o tamanho de cada sub-parte de $MSK$ gerada pelos nós. Como os nós devem estar próximos durante a inicialização do sistema, a contagem de saltos não é considerada.

### 5.4.2 Associação de novos membros ao D-PKG

Como descrito na seção 5.2.2, quando um dado nó $n_{new}$ deseja participar do D-PKG, ele deve contactar pelo menos $t$ membros do D-PKG ($\Omega$) solicitando autorização para atuar como membro do PKG. Cada um desses nós, se aceita $n_{new}$ como novo membro do D-PKG, calcula uma sub-parte de $MSK$ para $n_{new}$ e a envia para ele. Considerando um D-PKG com $m$ nós e a $MSK$ distribuída entre eles usando um esquema de limiar $(m, t)$, o custo para um novo membro participar do D-PKG, denotado por $NM$, é:

$$NM = (\Omega \ . \ sizeof(ReqMsg) + \Omega \ . \ sizeof(f^i_{new}(x))) \ . \ \Delta h \tag{5.10}$$

em que $ReqMsg$ é a mensagem enviada por $n_{new}$ para os nós de $\Psi$, $f^i_{new}(x)$ é cada sub-parte de $MSK$ enviada para $n_{new}$ e $\Delta h$ é a média de saltos entre os nós.

### 5.4.3 Emissão de chave privada dos nós

O custo de comunicação para cada nó recuperar a sua chave privada do D-PKG é igual ao custo para um novo membro se associar ao D-PKG. Como descrito na seção 5.2.3, cada nó deve contactar no mínimo $t$ membros do D-PKG, denotado como $\Psi$, solicitando a sua parte da chave privada. Cada nó de $\Psi$ envia a parte da chave privada ao nó requisitante. Assim, o custo de comunicação, denotado por $SKI$, pode ser definido como:

$$SKI = (\Psi \ . \ sizeof(ReqMsg) + \Psi \ . \ sizeof(\sigma^i_j)) \tag{5.11}$$

em que $ReqMsg$ é a mensagem enviada por $n_i$ aos membros de $\Psi$, $\sigma^i_j(x)$ é cada sub-parte de $SK_j$ enviada ao $n_j$.

### 5.4.4 Atualização de chaves

Como previamente descrito, qualquer nó pode atualizar localmente sua chave pública sem qualquer custo de comunicação adicional. Por outro lado, atualizar sua chave privada na fase $p_i$, ele precisa de $MSK_{p_i}$, que é emitida colaborativamente por $t$ membros do D-

PKG. Um dado nó, membro do D-PKG, seleciona pelo menos $t-1$ nós do D-PKG, denotado por $\Phi$, e requer a emissão de $MSK_{p_i}$. Todos os nós selecionados calculam suas partes de $MSK_{p_i}$ e a enviam para o nó solicitante. Esse nó, após obter $t$ partes, incluindo a parte gerada localmente, calcula $MSK_{p_i}$ e a envia para todos os nós não revogados usando uma difusão cifrada. Assim, o custo para atualizar essas chaves é único em cada fase. Ele pode ser definido como:

$$(\Phi \ . \ sizeof(ReqMsg) + \Phi \ . \ sizeof(RepMsg)) + BcastMsg \tag{5.12}$$

em que $ReqMsg$ é a mensagem enviada por um membro do PKG para os nós $\Phi$ solicitando a emissão de um novo elemento de chave privada comum, $RepMsg$ é a mensagem de resposta de cada nó de $\Phi$ e $BcastMsg$ é a mensagem de difusão cifrada enviada para todo os nós do sistema.

### 5.4.5 Revogação de chaves

O custo para revogar a chave privada de um dado nó $N_b$ depende do número de nós que consideraram $N_b$ comprometidos. Cada nó que detecta o mau comportamento de $N_b$ envia uma mensagem de acusação para todos os nós do D-PKG. Para considerar $N_b$ como comprometido, os membros do D-PKG devem receber pelo menos $\gamma$ acusações.

Após receber o número necessário de acusações, os membros do D-PKG trocam mensagens entre eles para emitir a mensagem de revogação. Então, a revogação de mensagem é enviada via difusão cifrada para todos os nós. Considerando $\gamma$ acusadores, o custo de revogação de chave:

$$(\gamma \times t) \ . \ sizeof(AcMsg) + (t)^2.sizeof(revMsg) + BcastMsg \tag{5.13}$$

em que $AcMsg$ é a mensagem de acusação enviada aos nós acusadores para os membros do D-PKG, $revMsg$ é a mensagem de revogação e $BcastMsg$ é a mensagem de difusão cifrada enviada a todos os nós.

## 5.5 Resultados das simulações

O desempenho, eficiência e resistência a ataques do *i*FUSO foi avaliado por meio de simulações usando o Network Simulator 3.16. Nas simulações, 50 nós usam IEEE 802.11 com a função de coordenação distribuída como protocolo de acesso ao meio. O modelo de propagação é a reflexão no solo em dois raios e o raio de comunicação é 250m. Os nós se movimentam em uma área de 500m x 100m, 1500m x 300m e 4500m x 900m, seguindo um modelo de mobilidade *waypoint* aleatório com velocidades máximas de 2m/s, 5m/s, 10m/s e 20m/s, com tempos máximos de pausa de 0s, 10s e 20s. As dimensões da rede variam para simular diferentes cenários de densidade dos nós. As simulações duram 2000 segundos e os resultados apresentados são a média de 35 simulações com 95% de intervalo de confiança. A Tabela 5.2 resume os parâmetros de simulação. Para melhorar a legibilidade das figuras, a escala para as redes de 500m x 100m e 1500m x 300m são as mesmas, mas para a rede de 4500m x 900m é diferente.

Tabela 5.2: Parâmetros das simulações.

| **Parâmetros** | **Valores** |
|---|---|
| Nós | 50 |
| Velocidade máxima ($m/s$) | 2, 5, 10, 20 |
| Tempo de pausa ($s$) | 0, 10, 20 |
| Área ($m$ x $m$) | 500x100, 1500x300, 4500x900 |
| Raio de comunicação ($m$) | 250 |

As seguintes quatro métricas são consideradas:

a. taxa de nós completos: porcentagem de nós que completaram suas operações no *i*FUSO;

b. sobrecarga de comunicação: número de mensagens enviadas para executar as operações;

c. média de atraso das operações completas: tempo médio em segundos para completar as operações de gerenciamento de chaves; e

d. atraso máximo das operações completas: tempo máximo em segundos para completar as operações de gerenciamento de chaves.

Nessa tese, apenas os cenários com tempo de pausa igual a 0 segundos são apresentados. Os demais cenários com tempo de pausa igual a 10 e 20 segundos possuem resultados similares com variações estatisticamente irrelevantes. Os gráficos da Figura 5.1 mostram a porcentagem de nós que completaram corretamente as operações de gerenciamento de chaves. Note que em cenários com 500x100m e 1500x300m todos os nós completaram com sucesso as suas operações. Esses resultados mostram a factibilidade do *i*FUSO mesmo em cenários com alta mobilidade.

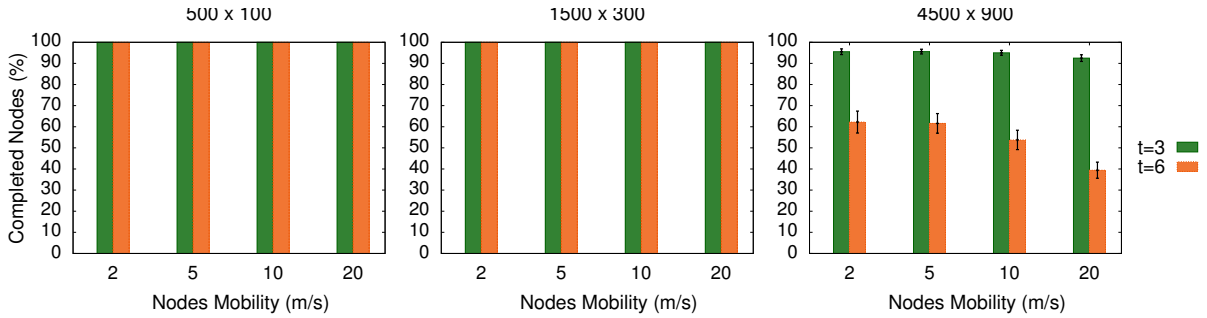



Figura 5.1: Taxa de nós completos.

Em cenários com 4500x900m, o sistema não é capaz de completar todas as suas operações. Esses resultados são consequência da baixa densidade da rede. Como os nós não são capazes de construir rotas funcionais, o *i*FUSO também não pode completar suas requisições. Note que com $t$ igual a 3, independente da mobilidade dos nós, mais de 90% dos nós completaram suas operações. Por outro lado, com $t$ igual a 6, a taxa de nós completo é próximo de 60% com nós se movimentando a 2m/s, 5m/s ou 10m/s. Com nós se movimentando a 20m/s, esse valor cai para 40%.

Os gráficos da Figura 5.2 ilustram a sobrecarga de comunicação, medida pelo número de mensagens necessárias para completar uma operação de gerenciamento de chaves. Nos cenários com 500x100m e 1500x300m a sobrecarga de comunicação é praticamente igual ao tamanho de $t$. Essa característica mostra que o *i*FUSO não onera a rede com retransmissões. Também, esse resultado é independente da mobilidade dos nós. Por outro lado, em cenários com 4500x900m, a sobrecarga de comunicação é um pouco maior. Nesse caso, independente da mobilidade dos nós e do tamanho de $t$, a sobrecarga de comunicação é próxima a 10 mensagens. Esse resultado também é resultante da baixa densidade da rede.

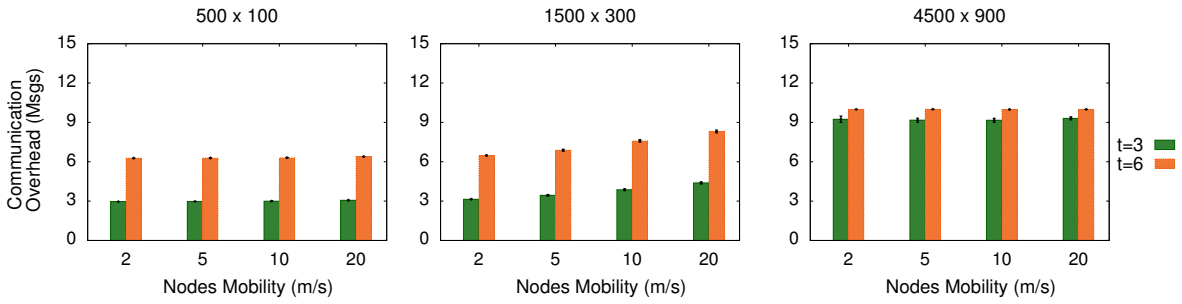



Figura 5.2: Sobrecarga de comunicação.

Os gráficos da Figura 5.3 mostram o atraso médio das operações completadas no *i*FUSO. Note que o atraso para completar as operações está diretamente relacionado com o tamanho da rede. Em cenários com 500x100m, os nós completam suas operações, na média, em menos de um segundo. Em cenário com 1500x300m e os nós se movimentando a 2m/s e 5m/s, o atraso médio é próximo a cinco segundos, independente de $t$. Já com os nós se movimentando a 20m/s o atraso médio está próximo de dez segundos com $t$ igual a 3 e vinte segundos com $t$ igual a 6. Em cenários com 4500x900m, em que nem todos os nós completaram as suas operações (Figura 5.1), o atraso médio é maior que 300 segundos em todos os cenários. Também o atraso médio é maior em cenários com baixa mobilidade, já que os nós não se movimentam muito pela rede, o que dificulta o contato com os membros do D-PKG.

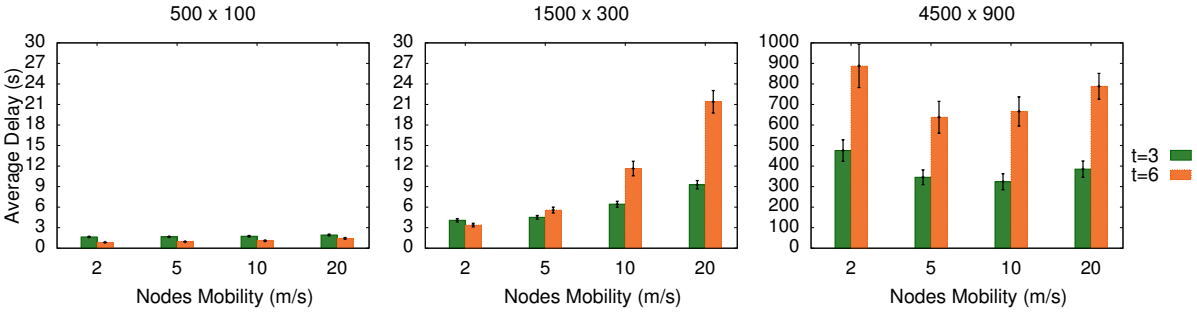



Figura 5.3: Atraso médio.

Finalmente, os gráficos da Figura 5.4 ilustram o atraso máximo para completar as operações do *i*FUSO. Em cenários com 500x100m e nós se movimentando a 20m/s, o atraso máximo não alcança dez segundos. Em cenários com 1500x300m e nós se movendo a 5m/s, esse valor é no máximo dez segundos para $t$ igual a 3 e vinte e cinco segundos

para $t$ igual a 6. Nesse cenário, o pior caso ocorre com nós se movimentando a 20m/s e $t$ igual a 6, quando o atraso máximo é de 80 segundos. Por outro lado, em cenários com área de 4500x900m, o atraso máximo é sempre maior que 500 segundos para $t$ igual a 3 e 1800 segundos para $t$ igual a 6. Esses resultados são a consequência da baixa densidade da rede, que afeta todos os protocolos de comunicação.

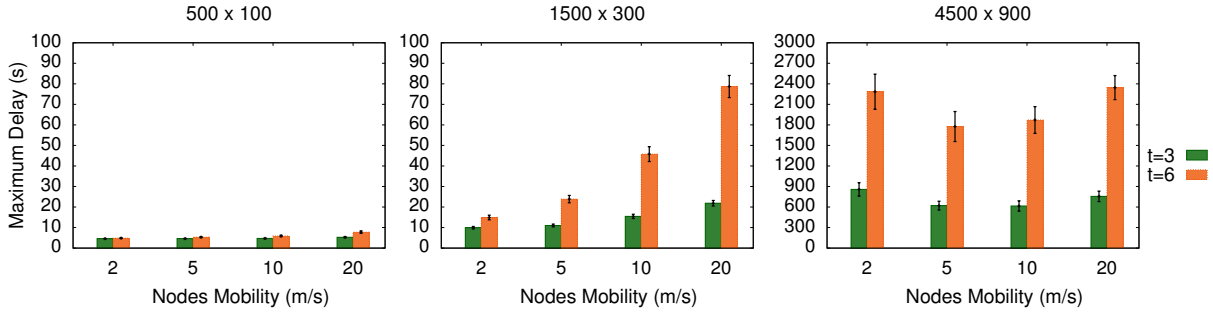



Figura 5.4: Atraso máximo.

## 5.6 Conclusão

Esse capítulo apresentou o *i*FUSO, uma solução totalmente auto-organizada e baseada em identidade para o gerenciamento de chaves, que pode ser utilizada no módulo de segurança do SEMAN. No *i*FUSO todas as operações são executadas pelos próprios nós de forma completamente distribuída, sem a necessidade de uma entidade centralizada. O esquema fornece mecanismos para revogar as chaves de nós maliciosos ou comprometidos, enquanto garante a renovação das chaves dos nós não comprometidos e legítimos. Ele também permite que os nós entrem e/ou saiam do sistema, mesmo que esses nós sejam participantes do D-PKG. Essas características não são encontradas em nenhuma outra solução baseada em identidade para as MANETs.

Os resultados das simulações mostram que as operações do *i*FUSO são efetivas e não impõem uma alta sobrecarga de comunicação. Quando uma rede é considerada conexa, todos os nós conseguem realizar as suas operações de gerenciamento de chaves com um baixo custo de comunicação.

# CAPÍTULO 6

# GERENCIAMENTO DE GRUPOS

Este capítulo apresenta como será realizado o gerenciamento de grupos para suporte às atividades do *middleware*. Nesta tese, um grupo é um conjunto de nós que compartilham interesses em comum e que desejam cooperar entre si na realização de atividades relacionadas a esse interesse. Esse 'interesse comum' é denominado *contexto*. Dessa forma, é importante que as informações de contexto sejam frequentemente atualizadas e estejam disponíveis, para que os grupos possam ser eficientemente organizados (COURAND et al., 2005).

Devido à grande variedade de serviços que podem ser fornecidas pelo *middleware*, muitos tipos de grupos distintos podem ser formados, com características diferentes de mobilidade, tempo de vida, modo de organização, políticas internas, regras de associação, entre outros. Contudo, independente das características do grupo, o sistema deve fornecer meios de gerenciamento para permitir a criação e a atualização dos grupos existentes e de seus respectivos perfis.

Para suportar os diversos tipos de aplicações, com maiores ou menores restrições de segurança, são previstas duas formas de gerenciamento de grupos: *yellow pages* e grupos fechados. Nos grupos do tipo *yellow pages*, também chamados de grupos abertos, são fornecidas primitivas para que os nós possam formar grupos livremente e disponibilizar serviços relacionados ao seu contexto. Por serem grupos abertos, eles não gerenciam internamente a confiabilidade dos seus membros. Assim, são indicados para serviços que requerem um menor nível de confiança ou quando as próprias aplicações são responsáveis por essa tarefa.

Os grupos fechados usam as informações do gerenciamento de confiança para a sua formação. Assim, todos os serviços fornecidos pelos membros de um grupo de contexto fechado atendem aos requisitos de segurança definidos no perfil do grupo. Também, é

possível realizar a comunicação segura interna entre os seus membros, e externa, quando nós externos desejam solicitar serviços aos membros de grupo.

A próxima seção descreve alguns trabalhos relacionados ao gerenciamento de grupo nas MANETs. Em seguida, é apresentado um modelo distribuído para armazenamento das informações sobre os grupos. Então, são descritas as duas abordagens de organização dos nós em grupos: abertos e fechados. Na descrição dos esquemas de gerenciamento de grupos é considerada a notação apresentada na Tabela 6.1.

Tabela 6.1: Notação do gerenciamento de chaves

| **Notação** | **Descrição** |
|---|---|
| $N_i$ | identificação do nó $i$ |
| $G_\alpha$ | identificação do grupo $\alpha$ |
| $\mathbb{G}_1$ | grupo aditivo cíclico de ordem prima $p$ |
| $\mathbb{G}_2$ | grupo multiplicativo cíclico de ordem prima $p$ |
| $\zeta$ | tamanho em *bit* de um texto plano |
| $e$ | um emparelhamento bilinear em que $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ |
| $H_1(x)$ | função *hash* em que $H_1(x) = \{0,1\}^* \rightarrow \mathbb{G}_1^*$ |
| $H_2(x)$ | função *hash* em que $H_2(x) = \mathbb{G}_2 \rightarrow \mathbb{Z}_p^*$ |
| $H_3(x)$ | função *hash* em que $H_3(x) = \mathbb{G}_2 \rightarrow \{0,1\}^\zeta$ |
| $N_\mathcal{F}$ | nós fundadores |
| $GEK_\alpha$ | chave de cifração do grupo $\alpha$ |
| $GDK_\alpha$ | chave de decifração do grupo $\alpha$ |
| $Sign_i$ | parte da assinatura de grupo chave mantida pelo nó $i$ |

## 6.1 Trabalhos relacionados

Em (LIU et al., 2005) é apresentada uma solução genérica para o gerenciamento de grupos. A formação dos grupos ocorre considerando atributos relevantes definidos na criação dele, como: localização, número máximo de saltos, confiabilidade, mobilidade, entre outros. Na solução proposta, os grupos possuem um líder que concentra as decisões. Caso um líder saia do grupo, outro nó pode ser selecionado para assumir essa função. O esquema é simples e não apresenta estratégias para garantir que o grupo formado contenha as características desejadas, dependendo da decisão do líder.

Outra estratégia de formação de grupos é apresentada em (AIKEBAIER; ENOKIDO;

TAKIZAWA, 2012) que adota a criptografia de chaves públicas na comunicação entre os nós. Quando um nó deseja formar um grupo, ele 'convida' apenas os nós com os quais possui uma confiança direta. Para adicionar novos membros, o nó que formou esse grupo solicita aos membros para que eles indiquem outros nós que eles confiam. Todas essas trocas de mensagens são realizadas usando mensagens cifradas, para garantir a integridade e confiabilidade.

A abordagem de criar grupos baseado no contexto dos nós é utilizada em (BOTTAZZI; MONTANARI; ROSSI, 2008), na apresentação de um ambiente chamado CAMPE. Nesse caso, os nós formam '*clusters*' considerando a proximidade, as propriedades de recursos e o padrão de mobilidade dos nós. O CAMPE utiliza o modelo de gerenciamento baseado em super-nós, no qual um nó, eleito pelos demais, atua como o líder (*clusterhead*) do grupo. A formação dos grupos e a eleição do líder são realizadas aplicando uma técnica da teoria de jogos, chamada de "Tomada de Decisão Multi-critério", que permite identificar o *trade-off* entre soluções possivelmente inconsistentes considerando diferentes critérios, como nível de bateria, padrão de mobilidade, atributos dos nós, entre outros.

Outro esquema que utiliza o conceito de '*clusters*' é apresentado em (RACHEDI et al., 2010). Nesta solução, cada agrupamento de nós deve possuir uma autoridade certificadora e uma autoridade registradora, que pode ser composta por um ou mais nós. Para proteger a autoridade certificadora contra ataques, é formada uma rede desmilitarizada dinâmica composta por nós que formam a autoridade registradora. A comunicação entre a autoridade certificadora e os demais nós considera o jogo não-cooperativo de soma não zero. Dependendo dos resultados do jogo, a autoridade certificadora pode decidir adicionar mais nós ao conjunto da autoridade registradora. Essa solução não é totalmente adequada para as MANETs pois os grupos formados são pouco dinâmicos e dependem de uma autoridade certificadora interna além de inserir uma camada de comunicação adicional entre os nós e a autoridade certificadora.

Em (MARUTA; OKADA, 2012) os autores propõem um esquema de formação de grupos dinâmicos usando a técnica de teoria dos jogos do dilema do prisioneiro. Antes da formação do grupo, cada jogador (nó) negocia um acordo de auto-associação que define

a estratégia do grupo cooperante. Os autores mostram que os grupos se formam após sucessivas negociações.

No entanto, nenhuma dessas abordagens considera a natureza dinâmica das MANETs que requer tanto a formação de grupos com menores restrições de segurança, como grupos mais restritos. Além disso, as abordagens apresentadas não consideram totalmente o contexto das aplicações que podem ser fornecidas pelos membros de um grupo e não empregam soluções descentralizadas para mitigar o impacto de ataques maliciosos.

## 6.2 Armazenamento das informações sobre os grupos existentes

No SEMAN, os grupos de contexto são considerados serviços que estão disponíveis na rede. As informações sobre a existência de grupos e suas caraterísticas principais precisam estar, de alguma forma, acessíveis aos nós que desejam participar desses grupos ou usufruir dos serviços que estão sendo fornecidos por eles. Para tal, é importante que o *middleware* forneça meios para que essas informações sejam gerenciadas e disponibilizadas durante o seu funcionamento. Várias arquiteturas foram propostas para organizar o fornecimento de serviços nas MANETs. Um estudo inicial sobre essas arquiteturas pode ser encontrado em (VERVERIDIS; POLYZOS, 2008). De modo geral, as propostas podem ser classificadas em arquiteturas com diretórios e arquiteturas sem diretórios.

Na primeira abordagem, as informações sobre os grupos estão armazenadas em um diretório, que pode ser centralizado ou distribuído. Os nós que armazenam informações sobre os diretórios são chamados de nós servidores. Nesse caso, sempre que um nó deseja fornecer um serviço, ele procura algum nó servidor e solicita a associação do seu serviço. Já um nó que deseja utilizar esse serviço, precisa apenas contactar o nó servidor e obter uma lista dos nós que estão fornecendo o serviço solicitado.

Na segunda abordagem, as informações sobre os grupos não estão armazenadas em um diretório, e devem ser propagadas ou solicitadas sempre que necessário. Assim, quando um nó deseja fornecer um serviço na rede, ele difunde essas informações pela rede, de forma que elas alcancem o maior número de nós possíveis. Para isso, podem ser utilizadas as técnicas de inundação global ou inundação controlada. No caso da inundação con-

trolada, a informação pode ser propagada somente por alguns saltos, de forma seletiva, probabilística, ou qualquer outro método que reduza o custo de uma inundação global. Da mesma forma, quando um nó deseja utilizar um serviço e ele não possui informações locais sobre esse serviço, ele solicita informações na rede, via inundação global ou controlada.

Não existe um consenso sobre qual dessas estratégias melhor atende aos requisitos das MANETs. A descoberta dos grupos é considerada boa quando apresenta uma alta disponibilidade, mantendo um baixo custo de comunicação e pequenos atrasos. Assim, se a rede possui poucas requisições de serviços, uma estratégia sem diretórios com consultas sob demanda seria mais indicada para o ambiente. Por outro lado, uma rede com muitos serviços sendo disponibilizados mas poucas consultas a esses serviços, geraria custos de comunicação desnecessários para manter as informações sobre tais serviços.

Qualquer uma dessas arquiteturas pode ser utilizada no SEMAN. Nesta tese, será considerado o uso de uma arquitetura com diretórios totalmente distribuídos. Quando um novo grupo é criado, as suas informações são disseminadas na rede. Todos os nós armazenam localmente as informações sobre os grupos. Assim, sempre que um nó precisar de informações sobre algum grupo, ele deve obter tais informações localmente, sem atrasos ou custos adicionais.

## 6.3 Yellow Pages

Uma primeira estratégia para a formação de grupos no SEMAN, totalmente aberta e dinâmica, é chamada de *Yellow Pages*. Esta técnica, baseada no funcionamento das páginas amarelas tradicionais, funciona como um diretório de serviços que são fornecidos na rede por meio do *middleware*. A formação dos grupos está diretamente relacionada a algum tipo de serviços que está sendo oferecido. Quando um nó deseja fornecer um serviço na rede, ele informa ao *middleware* o serviço que está fornecendo. Então, o *middleware* propaga essa informação na rede, para que todos os demais nós saibam sobre o serviço que está sendo fornecido.

Quando um outro nó deseja utilizar um serviço, ele solicita ao *middleware* uma lista dos nós que estão oferecendo o serviço desejado. Com base nesta lista, o nó pode optar

pelo uso ou não do serviço considerando informações do componente de gerenciamento de confiança.

Esse tipo de abordagem é importante quando se deseja fornecer serviços sem um alto grau de segurança. Qualquer nó pode participar livremente de um grupo e fazer parte da lista dos nós que estão fornecendo um dado serviço. As aplicações clientes podem, contudo, determinar o grau de confiança que desejam no serviço que está sendo oferecido. Assim, o *middleware*, com base nas informações fornecidas pelo gerenciamento de confiança, pode selecionar os nós mais confiáveis que estão oferecendo o serviço.

### 6.3.1 Formação de um grupo aberto

Antes de iniciar um grupo, um nó precisa certificar que não existe um outro grupo com as mesmas características que ele está propondo. Para isso, ele faz uma consulta ao seu diretório local. Caso já exista um grupo com as características propostas, ele se associa ao grupo (seção 6.3.2). Caso contrário ele precisa tomar as providências para a criação desse novo grupo.

Quando um nó deseja formar um novo grupo de contexto, ele inicialmente define todas as características principais desse grupo, como identificador, padrão de mobilidade, informações de contexto, tipo de serviço oferecido e nós iniciais. Outras informações podem ser adicionadas para facilitar o gerenciamento dos grupos. Em seguida, o nó que está criando grupo dissemina essas informações na rede, como já discutido na seção 6.2.

### 6.3.2 Entrada e saída de membros em um grupo aberto

Quando um nó deseja participar de um grupo aberto $G_\alpha$, ele precisa criar uma mensagem informando que ele está fornecendo os mesmos serviços como descrito no perfil do grupo $G_\alpha$. Em seguida, ele deve disseminar essa informação na rede, para que todos os demais nós estejam cientes que ele também está fornecendo tais serviços.

Assim, sempre que um nó deseja utilizar um serviço na rede, ele precisa apenas fazer uma consulta local ao seu diretório e verificar quais nós estão fornecendo o serviço desejado. Note que não existe uma estratégia para impedir a participação de nós nos grupos

abertos. Qualquer nó pode enviar uma mensagem informando que está participando desse grupo.

Da mesma forma, quando um nó deseja sair do grupo, ele deve apenas criar uma mensagem informando que está saindo do grupo e disseminar essa informação pela rede. Porém, como o envio dessa mensagem não é obrigatório ou os nós podem sair da rede de forma involuntária, é preciso alguma outra técnica para garantir a consistência dos nós que estão participando do grupo. Assim, o nó que criou o grupo ou, no caso de sua ausência, o nó mais antigo que está participando do grupo, pode fazer, em intervalos pré-determinados, consultas aos membros do grupo, verificando sua disponibilidade. Assim, ao final de um ciclo, uma lista dos nós indisponíveis é disseminada pela rede, informando quais os nós que não fazem mais parte do grupo. Como a consulta dos nós ainda ativos gera uma sobrecarga de comunicação ao sistema, é importante que o intervalo de verificação não seja muito pequeno, para impedir que essas consultas possam afetar o desempenho da rede.

### 6.3.3 Utilizando serviços dos grupos abertos

Os grupos abertos não fornecem métodos nativos de comunicação segura entre os membros ou para a solicitação de serviços. Como os nós podem participar livremente dos grupos, não existe um controle de associação de membros, o que dificulta o estabelecimento de chaves de grupo. No entanto, isso não impossibilita que os serviços fornecidos por meio de um grupo aberto exijam que as solicitações sejam realizadas por meio de mensagens cifradas e assinadas.

Quando um nó deseja solicitar algum serviço aos membros de um grupo aberto, ele faz a solicitação diretamente a esses nós, usando mensagens de *unicast* ou *multicast*. Caso ele deseje utilizar mensagem cifradas, ele pode utilizar as primitivas de segurança fornecidas pelo *middleware* para comunicação entre os nós, suportadas pelos componentes de operações criptográficas e gerenciamento de chaves.

## 6.4 Grupos Fechados

Enquanto alguns serviços podem ser suportados por um esquema de gerenciamento de grupos aberto, outros serviços requerem um tipo de gerenciamento mais controlado. Neste caso, o *middleware* disponibiliza às aplicações um serviço dinâmico de gerenciamento de grupos fechados. Esses grupos são formados com base no contexto, interesse e requisitos de segurança das aplicações.

Um exemplo de uso de grupos fechados no SEMAN é o esquema de gerenciamento de chaves descrito no capítulo 5 e em (SILVA; ALBINI, 2013), que requer um serviço altamente confiável e restrito quanto à participação nas atividades do grupo. Assim, uma das formas mais indicadas para a formação de grupos fechados é a utilizada no capítulo 5, em que cada grupo é uma implementação de um esquema de gerenciamento de chaves autônomo.

Esta seção descreve as operações para o gerenciamento de grupo fechados, como formação do grupo, associação de novos membros e exclusão de membros.

### 6.4.1 Formação de um grupo fechado

Como assumido previamente, a formação dos grupos é baseada no contexto das aplicações. Cada nó é capaz de promover a formação de um grupo, de forma autônoma, sem uma entidade central ou um gerente de grupo. Durante a formação de um grupo, o nó que está criando o grupo deve especificar apenas o perfil do grupo e os requisitos de segurança.

Um grupo pode ser formado por um conjunto de nós, com uma única asserção: que esses nós possam trocar informações de forma segura para inicializar o grupo. Com isso, para iniciar um grupo os nós devem determinar:

a. o tamanho $n$ do grupo e o limiar de segurança $t$;

b. $p$ e $q$: dois números primos grandes, sendo que $q$ divide $(p-1)$;

c. $\mathbb{G}_1$: um grupo aditivo cíclico de ordem $p$;

d. um gerador $g \in \mathbb{G}_1$;

e. $\mathbb{G}_2$: um grupo multiplicativo cíclico de ordem $p$;

f. o tipo de emparelhamento e garantir que exista um emparelhamento bilinear $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ para esse tipo escolhido;

g. $\mathcal{G} = \langle e, \mathbb{G}_1, \mathbb{G}_2 \rangle$.

A definição desses valores pode ser realizada conjuntamente pelo nós, usando alguma abordagem de acordo, ou ainda proposto por algum nó fundador aos demais nós deste grupo. Após essa etapa, cada nó fundador deve ter os seguintes elementos públicos:

a. os números primos $p$ e $q$;

b. o gerador $g$ e o grupo aditivo cíclico $\mathbb{G}_1$;

c. o grupo multiplicativo cíclico $\mathbb{G}_2$;

d. $\mathbb{Z}_q^*$: um corpo elíptico com ordem $q$;

e. $H_1(x)$: uma função *hash* em que $H_1(x) = \{0,1\}^* \rightarrow \mathbb{G}_1^*$; e

f. $H_2(x)$: uma função *hash* em que $H_2(x) = \mathbb{G}_2 \rightarrow \mathbb{Z}_p^*$.

Para inicializar o grupo, os nós devem gerar uma identificação pública desse grupo e uma assinatura. Essa assinatura é distribuída pelos membros desse grupo usando um esquema de criptografia de limiar $(m, t)$ entre os $m$ nós fundadores, da seguinte forma:

a. Cada nó $N_i$ escolhe uma função polinomial simétrica de duas variáveis $f_i(x, y)$ sobre $\mathbb{Z}_q^*$ em que as duas variáveis $x$ e $y$ devem ser de ordem máxima $t$. A função polinomial pode ser descrita como:

$$f_i(x, y) = \sum_{k=0}^{t} \sum_{j=0}^{t} a_{k,j}^i x^k y^j,$$

em que $a_{k,j}^i \in \mathbb{Z}_q^*$, $a_{k,j}^i = a_{j,k}^i$ e $a_{0,0}^i = z_i$

b. Cada nó $N_i$ calcula $f_l^i(x) = f_i(x, l)$ para todo nó $N_l$ pertencente aos nós fundadores, como:

$$f_l^i(x) = f_i(x, l) = \sum_{k=0}^{t} \sum_{j=0}^{t} a_{k,j}^i x^k l^j,$$

Então, $N_i$ envia de forma segura $f_l^i(x)$ para $N_l$.

c. Cada nó $N_i$ calcula a sua parte da assinatura $Sign_i$:

$$Sign_i = f_i(x) = \sum_{j=1}^{n} f_j^i(x) = \sum_{j=1}^{n} f_j(x, i) = f(x, i)$$

d. A assinatura $Sign$ não é conhecida por nenhum nó, mas é definida como:

$$Sign = \sum_{N_i \in m} Sign_i \bmod q$$

Cada nó $N_i$, após calcular a sua parte $Sign_i$, publica $g^{Sign_i}$. Quando os nós receberem $t$ partes, eles podem calcular a identidade de grupo como $ID = \sum_{i=1}^{t} g^{Sign_i}$. Após calculada, a identidade $ID$ do grupo pode ser publicada para todos os demais nós da rede.

Depois da formação do grupo, os nós que desejam colaborar em um contexto ou interesse específico, devem procurar um grupo e requisitar a sua participação nele. Como cada grupo é configurado com o seu perfil e requisitos de segurança, os próprios nós podem decidir se os grupos disponíveis atendem aos seus interesses.

### 6.4.2 Associação a um grupo fechado

Como descrito na seção 6.4.1, cada grupo fechado possui o seu perfil e requisitos. Assim, os próprios nós podem optar por participar de um grupo que atenda aos seus interesses. Se um nó $N_x$ deseja participar de um grupo $G_\alpha$ ele deve solicitar aos membros de $G_\alpha$ a autorização para participar das atividades deste grupo. Para poder participar de $G_\alpha$, $N_x$ precisa da aprovação de um número $t$ de membros.

Para que um nó $N_x$ possa participar do grupo $G_\alpha$, os seguintes passos devem ser realizados:

a. o nó $N_x$ escolhe $t$ nós do grupo $G_\alpha$, representados por $\Omega$;

b. o nó $N_x$ solicita a cada nó de $\Omega$ para ser aceito como membro do grupo $G_\alpha$;

c. cada nó $N_j \in \Omega$ envia uma parte da informação $f(j,k) = S_j^k$ ao nó $N_k$;

d. após receber $t$ respostas, $N_x$ pode calcular sua parte polinomial $Sign_x$ usando interpolação de Lagrange:

$$Sign_k = S_k(x) = \sum_{j=1}^{t} \lambda_j S_j^k = \sum_{j=1}^{t} \lambda_j f(j,k) = f(x,k)$$

Após calcular $Sign_x$, $N_x$ pode participar de todas as operações do grupo.

### 6.4.3 Exclusão de membros de grupos fechados

Quando um nó não atende mais aos requisitos de segurança e/ou confiança de um grupo, ele deve ter sua permissão de participação no grupo revogada. Para isso, são utilizadas mensagens assinadas de acusação e uma lista de associações revogadas. Quando um dado nó $N_x$ possui uma quantidade de acusações superior a um limite $\gamma$, ele tem a sua associação revogada. O valor de $\gamma$ é um parâmetro de cada grupo, definido em seu perfil de criação.

Quando um nó $N_a$, com base nas informações fornecidas pelo gerenciamento de confiança, acredita que o nó $N_x$ não atende mais aos requisitos do grupo, ele emite uma mensagem assinada de acusação e envia para todos os demais membros do grupo. Ele pode enviar essa mensagem usando mensagens de *unicast* ou um esquema de difusão cifrada, como apresentado na seções 5.2.4 e 5.2.5. Ao receber $\gamma$ acusações, cada nó membro do grupo cria um registro de revogação da associação e o armazena localmente em uma lista de associações revogadas.

Essa lista de associações revogadas pode ser disponibilizada publicamente pelos nós, para que os membros externos saibam que um dado nó $N_x$ não possui mais autorização de participação no grupo.

## 6.5 Comunicação de grupo segura

Para a comunicação de grupo de forma segura é proposto o uso de um protocolo de acordo de chaves de grupo. Esse tipo de protocolo permite que um grupo de usuários troque informações sobre um canal de comunicação inseguro e público e chegue a um acordo de uma chave secreta que é utilizada para derivar uma chave de sessão. Então, a chave de sessão pode ser utilizada para garantir requisitos como autenticação, confidencialidade e integridade.

A abordagem de acordo de chaves de grupo é atrativa para as redes dinâmicas porque não requer a presença de um controlador central ou líder. Nesse caso, todos os usuários no grupo geram a chave de sessão. Dessa forma, nenhum usuário pode controlar ou prever a chave de sessão. Esse tipo de abordagem tem sido amplamente empregado em aplicações distribuídas e colaborativas, como compartilhamento de arquivos, computação distribuída, conferências de áudio e vídeo, entre outros.

Diversas propostas para o estabelecimento de uma chave de sessão de grupo podem ser encontrados na literatura (AUGOT et al., 2005; JUNG, 2006; ZHANG et al., 2011). Qualquer esquema, que utilize uma abordagem baseada em identidade pode ser facilmente utilizado no SEMAN. Sem a perda da generalidade, assume-se o esquema proposto por Zhang *et al.* em (ZHANG et al., 2011). Uma grande vantagem deste esquema é que ele permite que membros de fora do grupo enviem mensagem cifradas aos membros do grupo. Isso facilita a solicitação segura de serviços a grupos fechados.

Para o funcionamento desse esquema é necessário que todos os nós do grupo sejam participantes do esquema do gerenciamento de chaves apresentado no capítulo 5. Assim, os membros do grupo já devem possuir a sua chave pública e respectiva chave privada, emitida pelo D-PKG.

No acordo de chaves, os nós membros de um grupo emitem mensagens assinadas. A junção de todas a mensagens assinadas emitidas pelos membros de um grupo $G_\alpha$ forma uma chave de cifração de grupo, denominada $GEK_\alpha$, que pode ser publicada livremente na rede. No entanto, somente os membros do grupo são capazes de derivar uma chave de decifração de grupo $GDK_\alpha$. As próximas subseções apresentam como funciona o acordo

de chaves e a geração das chaves de cifração e decifração de grupo.

### 6.5.1 Acordo

Um dado nó $N_i$, com chave privada $sk_i$ e participante do grupo $G_\alpha$, deve realizar os seguintes passos para realizar o acordo de chaves:

1) Escolher um número aleatório $\eta_i \in \mathbb{Z}_q^*$.

2) Calcular $r_i = g^{\eta_i}$.

3) Escolher um número aleatório $k \in \mathbb{Z}_q^*$.

4) Calcular $g_1 = g^k$.

5) Para todo $1 \leq j \leq n$, calcular $f_j = H_2(N_j)$.

6) Para todo $1 \leq j \leq n$, calcular $z_{i,j} = sk_i f_j^{\eta_i}$.

7) Publicar $\sigma_i = (r_i, \varrho_i, \{z_{i,j}\}_{j \in \{1,\ldots,n\}, j \neq i})$.

Nesse caso, $\varrho_i$ é a assinatura baseada em identidade sobre o valor $r_i$. O elemento $z_{i,i} = sk_i^{\eta_i}$ não é publicado, mas mantido em segredo pelo nó $n_i$.

### 6.5.2 Geração e uso da chave de cifração

Para obter uma chave de cifração de grupo, um nó primeiro verifica os $n$ pares de mensagens de assinatura $(r_1, \varrho_1), \ldots (r_n, \varrho_n)$. Se todas essas assinaturas forem válidas, então o nó calcula:

$$w = \prod_{i=1}^{n} r_i \quad \text{e} \quad Q = \hat{e}(\prod_{i=1}^{n} H_1(N_i), g_1)$$

Em seguida, ele configura a chave de cifração de grupo como $GEK = (w, Q)$. Para cifrar uma mensagem $m$, qualquer nó, membro ou não do grupo, gera um texto cifrado seguindo os seguinte passos:

1) seleciona $\rho \in \mathbb{Z}_q^*$;

2) calcula $c_1 = g^{\rho}$, $c_2 = w^{\rho}$ e $c_3 = m \oplus H_3(Q^{\rho})$;

3) gera o texto cifrado $c = (c_1, c_2, c_3)$.

Depois que o texto cifrado $c$ é gerado, ele pode ser enviado pela rede e somente os membros do grupo de destino podem decifrar a mensagem transmitida.

### 6.5.3 Geração e uso da chave de decifração

Cada nó $N_i$ verifica os $n$ pares de mensagens de assinatura $(r_1, \varrho_1), ...(r_n, \varrho_n)$. Se todas as assinaturas forem válidas, o nó $N_i$ calcula $GDK = \prod_{j=1}^{n} z_{j,i}$ e faz a seguinte verificação:

$$\hat{e}(gdk_i, g) \stackrel{?}{=} \hat{e}(f_i, w).Q$$

Se a equação estiver correta, o nó $N_i$ aceita a chave $GDK$ como chave de decifração do grupo. Caso contrário, ele aborta o procedimento.

Quando um no $N_i$, membro do grupo, receber uma mensagem cifrada $c = (c_1, c_2, c_3)$, ele utiliza a chave de decifração $GDK$ para decifrar a mensagem, como segue:

$$m = c_3 \oplus H_3(\hat{e}(GDK, c_1)\hat{e}(f_i^{-1}, c_2))$$

## 6.6 Conclusão

Esse capítulo apresentou as abordagens de gerenciamento de grupos do SEMAN. Foram propostas duas abordagens para organização dos nós em grupos de contexto. A primeira permite a associação livre dos nós dentro dos grupos, chamados de grupos abertos ou *Yellow Pages*. Nesse caso, todo o controle de confiabilidade dos nós que estão oferecendo serviços nesses contextos, se necessário, deve ser realizado pelas próprias aplicações que utilizam o *middleware*. Por outro lado, esse tipo de abordagem impõe um baixo custo de gerenciamento para o sistema. Note que essa primeira abordagem não apresenta um aumento na segurança do sistema contra ataques maliciosos.

A segunda propõe a organização dos nós em grupos fechados, de acordo com o contexto

dos serviços que estão sendo providos. Nesse caso, são definidos, na criação dos grupos, os requisitos de segurança e confiança que cada nó deve ter para poder participar das atividades desse grupo. O esquema também fornece formas de revogar a participação que não atendem mais aos requisitos de segurança do grupo de contexto. Com isso, o *middleware* tem a sua resistência contra ataques bizantinos aumentada, pois os nós maliciosos não poderão realizar atividades em nome dos demais membros do grupo.

Também, o sistema torna-se menos susceptível a ataques de egoísmo, visto que são empregadas técnicas de compartilhamento $t$-sobre-$n$ na criação dos grupos. Assim, não é necessária a participação de todos os participantes de um grupo na prestação dos serviços fornecidos pelo grupo.

Por fim, foi apresentada uma técnica de comunicação segura de grupo, em que os nós que não são membros de um grupo de contexto fechado são capazes de enviar mensagens cifradas para os membros de um grupo fechado. O sistema também garante que somente os membros desse grupo serão capazes de decifrar a mensagem transmitida. Com isso, esse serviço aumenta a segurança do sistema contra ataques de personificação e *Sybil*.

# CAPÍTULO 7

# INTEGRAÇÃO DOS COMPONENTES EM CENÁRIOS DIVERSOS

O componente de gerenciamento de políticas tem como função auxiliar o módulo de segurança na integração dos seus componentes de gerenciamento de confiança, chaves e grupos. Assim, é fundamental que existam estratégias elaboradas para fornecer segurança nos mais distintos cenários em que as aplicações possam ser fornecidas usando o SEMAN.

Diante disso, este capítulo discute alguns estudos de caso, que mostram como o *middleware* pode ser utilizado em ambientes diversos. Os parâmetros de segurança que são descritos para cada um dos cenários apresentados são configurados no componente de políticas de segurança, parte integrante do módulo de segurança. É importante ressaltar que vários cenários podem ser encontrados em uma única rede. Algumas aplicações podem se adaptar melhor em cenários mais abertos, enquanto outras necessitam de um controle maior de segurança. O SEMAN possibilita a configuração desses diferentes cenários, pois as aplicações e os nós são organizados em contextos, com perfis e parâmetros de segurança configurados de acordo com os serviços que são fornecidos.

São apresentados três tipos de cenários:

**aberto:** indicado para aplicações que não necessitam de um grande controle de segurança ou que não comprometem todo o funcionamento da rede em caso de alguma falha;

**parcialmente restrito:** indicado para aquelas aplicações que necessitam de um suporte de segurança intermediário, porém não requerem um controle muito rígido das suas operações; e

**restrito:** indicado para as aplicações que requerem um alto grau de segurança para suportar as suas operações, ou aquelas aplicações que, se comprometidas, podem

afetar todo o sistema.

Para cada um dos cenários apresentados são indicadas políticas de segurança distintas para os componentes que são fornecidos pelo módulo de segurança. A Figura 7.1 ilustra como os componentes de segurança podem ser configurados para se adaptarem aos cenários propostos. Contudo, essa lista de cenários e políticas não é estática, e novos ambientes e configurações podem ser idealizados e configurados pelos usuários do *middleware*.

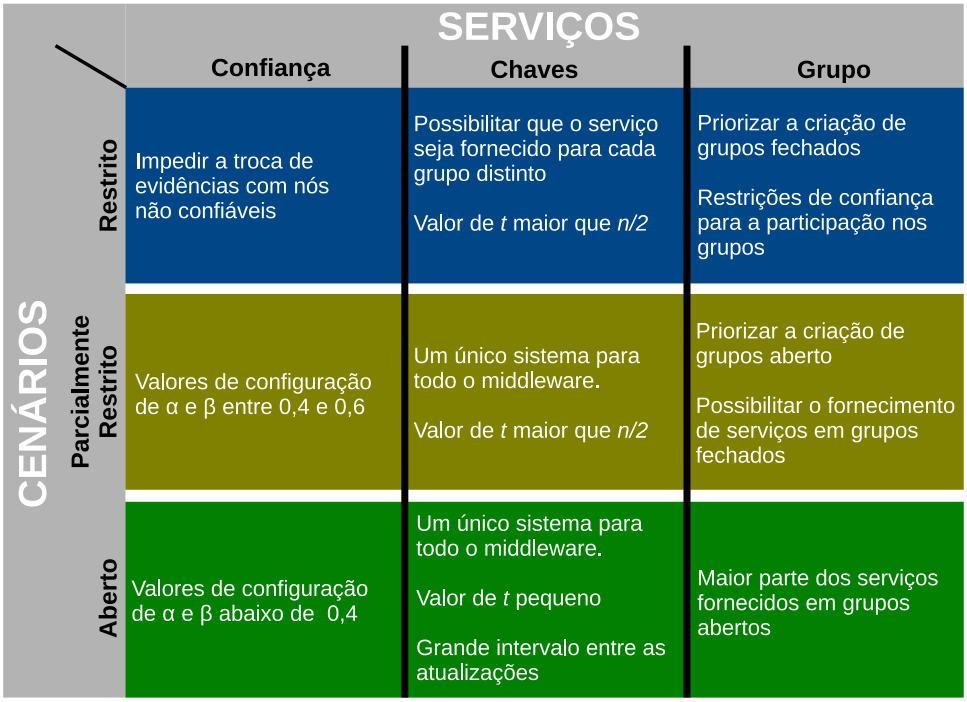



Figura 7.1: Políticas de segurança para cenários distintos.

As próximas seções detalham esses cenários e como os componentes de segurança podem ser integrados no fornecimento de serviços às aplicações. Em cada cenário é discutido como o *middleware* pode garantir a segurança desejada e qual a sobrecarga de comunicação que os componentes de segurança impõem ao sistema. Contudo, medir esse tipo de sobrecarga nos serviços fornecidos pelo SEMAN é uma tarefa complexa, pois esse custo depende de muitos fatores, como: tamanho dos grupos, intervalo de atualização dos componentes, valores de limiar $t$ dos serviços, entre outros. Portanto, uma aproximação mais realista da sobrecarga é considerada um trabalho futuro.

## 7.1 Cenários abertos

Um primeiro cenário que o *middleware* pode ser empregado é um ambiente mais aberto, em que é necessário um controle de segurança menos rigoroso. Diversas aplicações podem querer fornecer serviços em um cenário aberto. Um exemplo seria um serviço de compartilhamento distribuído de dados e arquivos. Nesse caso, os usuários poderiam compartilhar, temporariamente, dados ou arquivos que não requerem um alto grau de sigilo e disponibilidade, como arquivos de vídeo ou áudio. Assim, a aplicação precisa de um *middleware* que possibilite o agrupamento dos usuários, e até mesmo o gerenciamento de confiança dos serviços que estão sendo fornecidos. Porém, ela não requer um alto grau de confiabilidade dos nós para que eles possam passar a fornecer esse tipo de serviço. Assim, os parâmetros e limites dos componentes de segurança podem ser configurados com poucas restrições. A seguir, seguem algumas sugestões:

a. **gerenciamento de confiança:** sendo adotado o TRUE para essa atividade, os valores de $\alpha$ e $\beta$ podem possuir valores baixos, menores de 0,4. Com isso, o *middleware* irá considerar mais nós confiáveis no contexto que está sendo avaliado. Como resultado direto, mais nós poderão ser considerados confiáveis nas avaliações deste contexto;

b. **gerenciamento de chaves:** com o *i*FUSO, os valores de $t$ para o compartilhamento da chave mestre podem ser pequenos, menores que $n/2$, por exemplo. Além disso, o tempo entre as fases de atualização pode ser grande. Com isso, a sobrecarga do sistema é reduzida, enquanto o serviço é oferecido aos usuários dentro dos parâmetros determinados; e

c. **gerenciamento de grupos:** em cenários abertos, as aplicações podem ser fornecidas em grupos abertos, sendo que os próprios usuários das aplicações podem consultar o componente de gerenciamento de confiança para optarem pelo uso ou não dos serviços oferecidos por um nó.

Note que, neste cenário, pode ser usado um único grande grupo para gerenciamento de chaves de todo o *middleware*. Com isso, todas as aplicações que precisem dos serviços

de criptografia usam o mesmo serviço. Para isso, o gerenciamento de confiança deve possuir informações de um contexto que será consultado pelo gerenciamento de chaves, por exemplo key-management.

Para todos os demais serviços fornecidos dentro da rede, o gerenciamento de confiança pode prover informações de confiança no contexto desses serviços. Por exemplo, um serviço de localização de recursos possui um contexto diferente de um serviço de armazenamento distribuído. Os nós que fornecem esses serviços podem estar organizados em grupos abertos, mas os seus usuários podem utilizar os valores de confiança fornecidos pelo *middleware* para escolherem os servidores que melhor se adaptem aos seus requisitos.

A Figura 7.2 ilustra uma forma de configuração do módulo de segurança do SEMAN para atender aplicações em um cenário aberto. Nesse caso, a aplicação 'compartilhamento de dados' solicita serviços ao *middleware*. Os módulos de serviços e processamento atendem aos pedidos da aplicação e, sempre que necessário, fazem consultas ao módulo de segurança. Neste cenário, o gerenciamento de confiança provê dois contextos: 'sharing' e 'KeyManagement'. Ambos possuem valores de $\alpha$ e $\beta$ abaixo de 0,4.

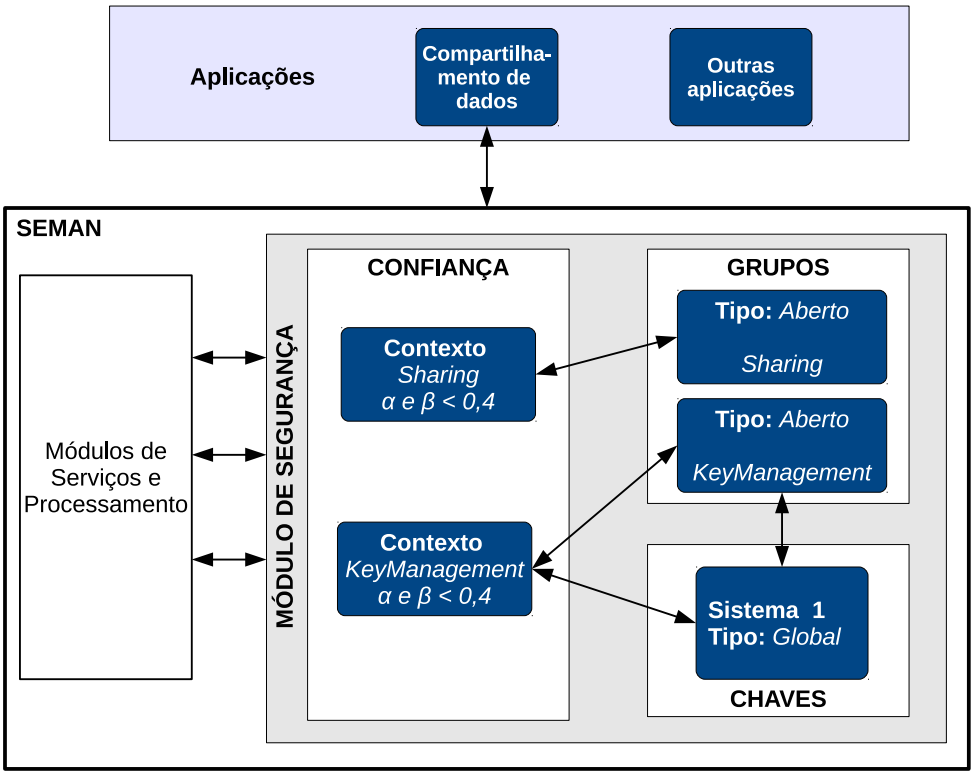



Figura 7.2: Cenário Aberto.

O contexto KeyManagement é consultado pelo sistema de gerenciamento de chaves para a emissão, revogação e atualização de chaves. Contudo, embora sejam considerados os valores do gerenciamento de confiança para a emissão das chaves, os limiares para a aceitação são pequenos. Assim, qualquer nó pode pedir a participação como membro no D-PKG, tornando o grupo aberto, mas a emissão da parte da chave mestre privada do sistema é emitida apenas se esse nó atende aos requisitos mínimos de confiança.

O outro contexto é chamado de Sharing e pode ser consultado pelas próprias aplicações na aceitação ou não dos serviços fornecidos pelos membros de um grupo aberto chamado 'Sharing'. Note que qualquer nó pode participar desse grupo aberto. Se necessário, os membros do grupo ou as aplicações clientes podem consultar o sistema global de gerenciamento de chaves para confirmar a autenticidade da identidade de algum outro membro do grupo.

Com essas configurações o *middleware* fornece poucas garantias de segurança às aplicações. As aplicações que realizarem consultas ao componente de gerenciamento de confiança tendem a receber informações imprecisas, visto que sistema torna-se susceptível a ataques de falsa acusação. Já o gerenciamento de chaves pode sofrer com ataques de personificação, pois um atacante precisa comprometer um número menor de membros do D-PKG para ser aceito e receber a sua chave privada ou até mesmo ser membro do próprio D-PKG.

## 7.2 Cenários parcialmente restritos

Um segundo cenário é um ambiente parcialmente restrito, em que é necessário um controle de segurança intermediário. Um exemplo de serviço que pode ser classificado como parcialmente restrito é a localização de recursos. Nesse caso, é muito importante que o *middleware* forneça garantias da autenticidade dos nós que estão oferecendo esse serviço, mas, ao mesmo tempo, não precisa impedir que qualquer nó ofereça um serviço às demais aplicações.

Nesse caso, os parâmetros e limites dos componentes de segurança podem ser configurados com mais restrições que no cenário anterior. A seguir, seguem algumas sugestões:

a. **gerenciamento de confiança:** sendo adotado o TRUE para essa atividade, os valores de $\alpha$ e $\beta$ podem possuir valores em torno de 0,5 e 0,7. Com isso, o *middleware* irá trocar informações com os nós que possuem uma avaliação de confiança intermediária no contexto que está sendo avaliado. Como resultado direto, o esquema terá uma sobrecarga menor que o cenário anterior, enquanto tem um controle um pouco maior das informações trafegadas;

b. **gerenciamento de chaves:** com o *i*FUSO, o valor de $t$ para o compartilhamento da chave mestre deveria ser maior que $n/2$, para prevenir ataques de particionamento do sistema de gerenciamento de chaves. Além disso, o tempo entre as fases de atualização das chaves não pode ser muito grande, para impedir que nós com baixa confiança permaneçam muito tempo no sistema; e

c. **gerenciamento de grupos:** nesse cenário, algumas aplicações podem ser fornecidas em grupos fechados, mas a maior parte pode ser organizada em grupos abertos. Assim, embora todos os nós possam se associar a um grupo e fornecer serviços neste contexto, as aplicações clientes podem utilizar os dados do gerenciamento de confiança para decidirem qual o melhor nó para solicitarem o serviço.

Note-se que para esse tipo de cenário, ainda pode ser usado um único grande grupo para gerenciamento de chaves de todo o *middleware*. Com isso, todas as aplicações que precisem dos serviços de criptografia usam o mesmo serviço. Para isso, como nos cenários abertos, o gerenciamento de confiança deve possuir informações de um contexto que será consultado pelo gerenciamento de chaves, por exemplo key-management. Para todos os demais serviços fornecidos dentro da rede, o gerenciamento de confiança pode prover informações de confiança no contexto desses serviços.

A Figura 7.3 ilustra como os componentes do SEMAN podem ser integrados para atenderem aos requisitos de segurança em cenários parcialmente restritos. A aplicação 'localização de recursos' solicita serviços ao *middleware*, que são atendidos pelos módulos de serviços e processamento. Quando necessário, são realizadas consultas ao módulo de segurança. São previstos dois contextos: KeyManagement e Localizacao. Em ambos os

contextos, o componente de gerenciamento de confiança possui os valores de $\alpha$ e $\beta$ em torno de 0,5 e 0,7.

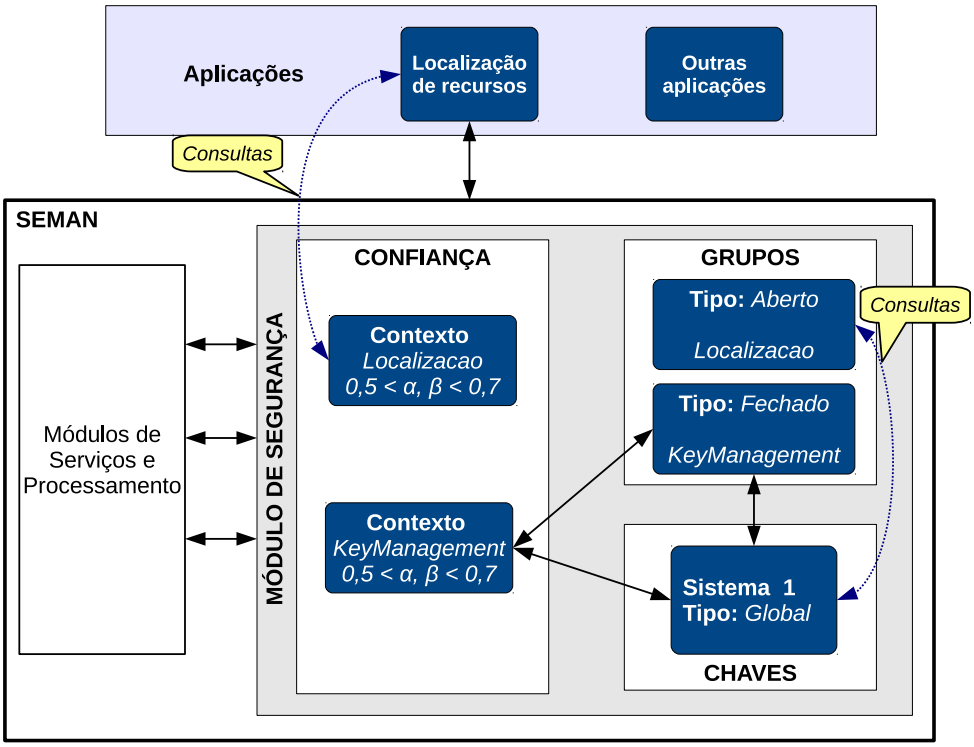



Figura 7.3: Cenário parcialmente restrito.

Como nos cenários abertos, o contexto KeyManagement é consultado pelo sistema de gerenciamento de chaves para a emissão, revogação e atualização de chaves. Contudo, são aceitos como membros do D-PKG apenas os nós que fazem parte do grupo fechado denominado 'KeyManagement'. Assim, a participação de nós como membros do D-PKG torna-se mais restrita, dando maior confiabilidade ao sistema. Também, os parâmetros $\alpha$ e $\beta$ do contexto KeyManagement podem ser aumentados, para dar maior segurança ao gerenciamento de chaves, e consequentemente, às aplicações clientes do *middleware*.

O outro contexto é chamado de Localizacao e pode ser consultado pelas próprias aplicações na aceitação ou não dos serviços fornecidos pelos membros de um grupo aberto chamado 'Localizacao'. Como nos cenários abertos, qualquer nó pode participar desse grupo aberto e, se necessário, os membros do grupo ou as aplicações clientes podem consultar o sistema global de gerenciamento de chaves para confirmar a autenticidade da identidade de algum outro membro do grupo.

Com essas configurações o sistema estaria protegido contra os ataques de personifi-

cação. Também, valores maiores de $\alpha$ e $\beta$ dificultam a propagação de falsas acusações contra a reputação dos nós dentro dos contextos avaliados. Assim, embora os serviços estejam sendo oferecidos, em grande parte, em grupos abertos, os usuários desses serviços recebem a garantia do *middleware* da autenticidade dos nós que estão participando dos grupos.

## 7.3 Cenários restritos

Um terceiro cenário é um ambiente restrito, em que é necessário um controle de segurança muito rigoroso. Nesse tipo de cenário estão as aplicações que não podem ser comprometidas em caso de ataques. Essas aplicações geralmente realizam tarefas fundamentais aos seus usuários e, se afetadas por ataques maliciosos, podem comprometer a integridade dos serviços fornecidos. Um exemplo seria um serviço de transações comerciais ou financeiras, que não pode ser afetado por ataques maliciosos. Esses serviços devem receber a garantia do *middleware* de que estão protegidos contra a ação maliciosa dos atacantes.

Nesse caso, os parâmetros e limites dos componentes de segurança devem ser configurados com muitas restrições. A seguir, seguem algumas sugestões:

a. **gerenciamento de confiança:** sendo adotado o TRUE para essa atividade, os valores de $\alpha$ e $\beta$ devem possuir valores altos, maiores que 0,7. Com isso, o *middleware* irá trocar informações apenas com os nós confiáveis no contexto que está sendo avaliado. Como resultado direto, menos nós poderão participar de grupos deste contexto e das suas atividades;

b. **gerenciamento de chaves:** com o $i$FUSO, os valores de $t$ para o compartilhamento da chave mestre devem ser maiores que $n/2$. Além disso, o tempo entre as fases de atualização das chaves não pode ser grande. Com isso, a sobrecarga do sistema é aumentada, mas os serviços fornecidos aos usuários irão garantir segurança contra ataques maliciosos; e

c. **gerenciamento de grupos:** em cenários restritos, as aplicações devem ser fornecidas em grupos fechados, com o controle para a participação no grupo bem restrito.

Note que para esse tipo de cenário, podem ser utilizados diferentes grupos de gerenciamento de chaves, dependendo das aplicações. Dessa forma, as aplicações mais restritas devem possuir o seu próprio gerenciamento de chaves dentro de seu contexto. Isso impede que nós maliciosos, mesmo sem participarem do grupo que está fornecendo o serviço, possam, por exemplo, serem membros do D-PKG. Por outro lado, pode existir um esquema de gerenciamento de chaves global para fornecer serviços aos grupos com aplicações menos restritas. Para todos os demais serviços fornecidos dentro da rede, o gerenciamento de confiança pode prover informações de confiança no contexto desses serviços. Essas configurações garantem às aplicações a formação de grupos de contexto com nós que atendem aos requisitos de alta confiabilidade necessários neste cenário.

A Figura 7.3 ilustra a integração dos componentes do SEMAN no fornecimento de serviços à aplicações de cenários restritos. A aplicação 'transações comerciais' solicita serviços ao *middleware*. Esses pedidos são atendidos pelos módulos de serviços e processamento que, quando necessário, consultam o módulo de segurança. São previstos dois contextos: DPKG-Transacoes e Transacoes. Em ambos os contextos, o componente de gerenciamento de confiança possui os valores de $\alpha$ e $\beta$ maiores que 0,7.

O contexto DPKG-Transacoes é consultado pelo sistema de gerenciamento de chaves formação dos D-PKG que será responsável pelo gerenciamento de chaves do grupo fechado 'Transacoes'. Para isso, o componente de gerenciamento de grupos possibilita a criação de um grupo fechado denominado 'DPKG-Transacoes', que contém apenas os nós que atendem aos requisitos para serem membros do D-PKG. Esses membros do D-PKG fazem consultas ao grupo fechado 'Transacoes' para verificar a confiabilidade dos nós e emitir as chaves privadas para eles.

O outro contexto é chamado de Transacoes e é consultado pelos membros do grupo fechado de mesmo nome, que decidem pela aceitação ou não de um novo membro no grupo fechado, ou pela exclusão de um membro. Diferente dos cenários anteriores, para participarem de um grupo, os nós precisam atender aos requisitos de confiança definidos

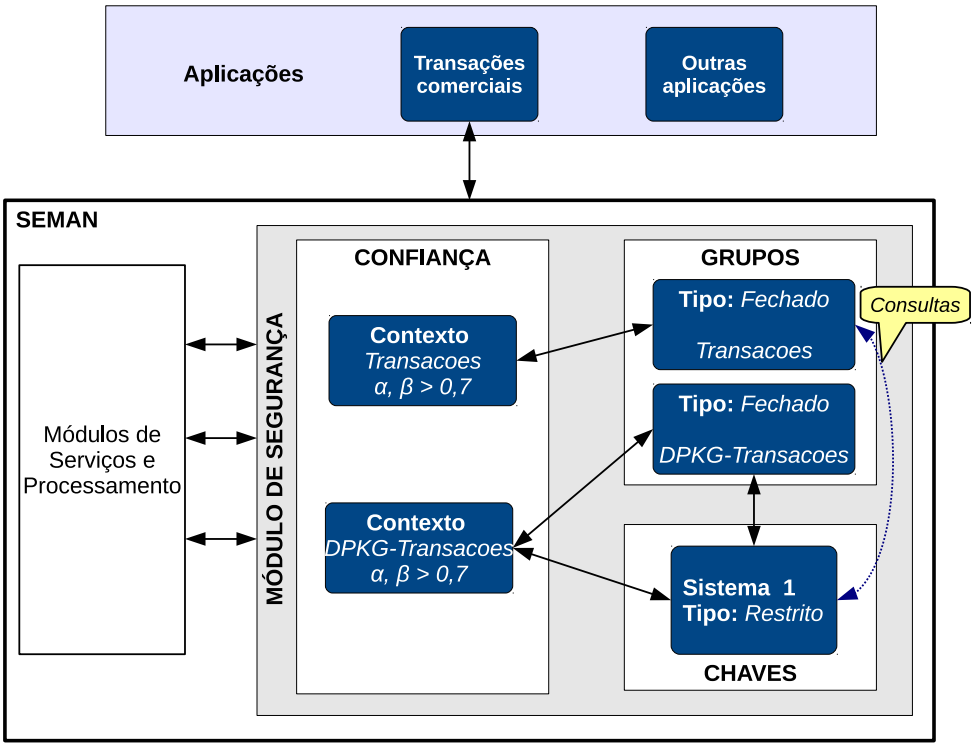



Figura 7.4: Cenário restrito.

pelas políticas do grupo, o que aumenta a segurança do sistema contra ataques maliciosos.

## 7.4 Cenários híbridos

Foram apresentados três cenários distintos de configuração do módulo de segurança do SEMAN. Contudo, na prática, cada aplicação que esteja utilizando os serviços do *middleware* pode apresentar um cenário diferente. Por exemplo, ao mesmo tempo, o *middleware* pode fornecer serviços a aplicações que requerem um alto nível de segurança e também a outras aplicações que são mais abertas. Dessa forma, as políticas de segurança do SEMAN devem ser direcionadas aos contextos das aplicações e serviços que utilizam o módulo de segurança do *middleware*.

Uma recomendação para o uso do SEMAN nesses cenários é a configuração de um gerenciamento de chaves global, que atenda a todos os serviços suportados pelo *middleware*. Assim, toda a rede é atendida por um único D-PKG e todos os usuários possuem um único par de chave pública e privada que servem para utilizar todas as aplicações. O gerenciamento de confiança deve oferecer para o gerenciamento de chaves informações

sobre o usuários em dois contextos: key-management e key-management-dpkg.

As informações de confiança sobre o primeiro contexto (key-management) são utilizadas pelo gerenciamento de chaves na decisão pelos membros do D-PKG pela emissão, atualização ou revogação das chaves privadas dos usuários. Nesse caso, os usuários que possuem um valor de confiança neste contexto abaixo de um limiar pré-determinado (que deve ser um valor alto), não terão suas chaves privadas emitidas ou atualizadas pelos membros do D-PKG.

Já as informações de confiança do segundo contexto (key-management-dpkg) são utilizadas pelos membros do D-PKG na decisão de aceitação ou não de um novo membro ao D-PKG. Essas restrições devem ser maiores que as de emissão da chave privada. Por exemplo, um nó pode ser autêntico e ter o direito a ter a sua chave privada emitida pelo D-PKG, no entanto, pode não ser confiável o bastante para ser parte do PKG distribuído que emite novas chaves aos usuários.

Com um sistema de gerenciamento de chaves seguro, o *middleware* garante a proteção das operações criptográficas contra ataques maliciosos e a autenticidade dos participantes. Assim, cada aplicação pode ser fornecida dentro de um contexto de grupos abertos ou fechados, que requerem um alto grau de confiabilidade ou que permitem que os serviços sejam fornecidos por qualquer nó da rede. Com isso, cada aplicação estará protegida contra a ação maliciosa dos nós dependendo das suas políticas de aceitação de serviços por parte de nós mais ou menos confiáveis.

## 7.5 Conclusão

Este capítulo apresentou um estudo de alguns cenários que o *middleware* SEMAN pode ser utilizado para atender aos requisitos das aplicações. Foram apresentados três cenários distintos: aberto, parcialmente restrito e restrito. Em cada um desses cenários foram discutidas formas de se configurar os componentes de segurança para suportar as necessidades das aplicações. Também foram discutidas as proteções garantidas pelo *middleware* nesses cenários e a sobrecarga de comunicação esperada.

Além disso, foi discutido um cenário híbrido, comum nas redes reais, em que diversos

contextos, com perfis de segurança diferentes podem ser necessários em uma rede. Nesse caso, foram recomendados parâmetros de configuração em que todos os serviços são suportados por um único sistema de gerenciamento de chaves global, que tem como função garantir a segurança das operações criptográficas do SEMAN. Todos os demais serviços podem ser organizados em grupos de contexto distintos, abertos ou fechados, mas usufruem dos serviços do gerenciamento de chaves global. Contudo, isso não impede que, se necessário, um contexto ofereça um serviço de gerenciamento de chaves independente.

A Figura 7.5 ilustra a sobrecarga esperada em cada um dos cenários apresentados. O *middleware*, por gerar um controle de segurança maior, implica em alguma sobrecarga de comunicação à rede. Contudo, o SEMAN não implica um alto custo de comunicação. O gerenciamento de chaves, por exemplo, necessita de maiores trocas de mensagens na criação do grupo e mensagens localizadas na emissão da chave privada dos nós. O seu maior custo acontece nas atualizações de chaves, que requerem mais trocas de mensagens. Contudo, essas atualizações não acontecem constantemente, o que não afeta o desempenho da rede.

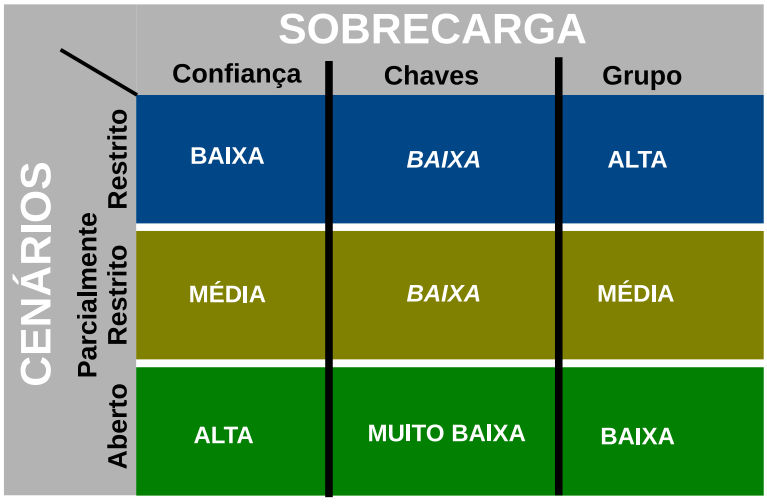



Figura 7.5: Sobrecarga de comunicação esperada nos cenários.

O gerenciamento de grupos possui um custo que depende muito da forma como os grupos estão organizados. Uma rede com diversos grupos fechados, tende a ter uma sobrecarga de comunicação maior do que a rede com grupos abertos. No entanto, mesmo em cenários com grupos fechados, esse custo maior de comunicação acontece apenas na cria-

ção dos grupos. Além disso, o esquema de comunicação segura em grupo permite que as mensagens enviadas ao grupo sejam transmitidas via *multicast*, diminuindo a quantidade de mensagens individuais, quando os grupos possuem muitos membros.

Por fim, o gerenciamento de confiança implica em um custo maior quando os grupos são mais abertos e, com isso, os valores de $\alpha$ e $\beta$ são menores. Contudo, mesmo essa maior quantidade de troca de mensagens é realizada apenas entre os nós vizinhos, afetando pouco o restante da rede.

Com isso, o módulo de segurança do SEMAN pode ser utilizado para garantir os requisitos de segurança das aplicações, enquanto não impõe uma alta sobrecarga de comunicação ao sistema.

# CAPÍTULO 8

# CONCLUSÃO

Este capítulo apresenta as principais contribuições do trabalho, as publicações realizadas durante o seu desenvolvimento e as sugestões de trabalhos futuros.

## 8.1 Considerações finais

As MANETs são redes formadas espontaneamente, caracterizadas pela ausência de uma infraestrutura fixa e controle centralizado. Tais redes são construídas por unidades móveis e possuem uma topologia dinâmica. Essas características as tornam altamente atrativas em cenários que exigem o desenvolvimento rápido da rede e que possuem dificuldade de estabelecimento de infraestrutura. Por outro lado, também as tornam vulneráveis a ataques. Porém, as suas aplicações requerem serviços confiáveis e seguros.

Para suportar as aplicações distribuídas nas MANETs, diversos *middlewares* foram apresentados e discutidos. Eles estão classificados, neste trabalho, em baseados em espaço de tuplas, baseados em P2P, baseados em contexto, *cross-layer* e orientados à aplicação. De modo geral, eles endereçam os problemas clássicos das MANETs, como a necessidade de comunicação assíncrona. Contudo, eles não consideram, ou apenas abordam superficialmente, as questões de segurança que afetam a confiabilidade dessas redes.

Este trabalho propôs o desenvolvimento de um *middleware* seguro baseado em contextos, chamado de SEMAN, e que utiliza uma abordagem de grupos como suporte às tomadas de decisão quanto a segurança. Foi discutida a arquitetura do *middleware* e a visão geral do seu funcionamento. Também foi apresentado como os serviço do SEMAN devem ser fornecidos e como a abordagem de grupos pode ser aplicada para garantir a segurança na comunicação e no fornecimento destes serviços.

O SEMAN possui três módulos: serviços, processamento e segurança. Os dois primeiros são responsáveis pelo fornecimento de serviços e pelo gerenciamento dos pedidos

que são requisitados ao *middleware*. O módulo de segurança tem a função de garantir a segurança às aplicações que utilizam os serviços fornecidos pelo SEMAN. Este módulo é composto pelos componentes de gerenciamento de confiança, chaves e grupos. Todos esses componentes foram detalhados, enfatizando como eles podem utilizados para garantir a segurança às aplicações.

Os componentes do módulo de segurança são integrados pelo gerenciamento de políticas que é responsável pelos parâmetros de segurança de cada serviço fornecido pelo *middleware*. Além disso, todas as atividades são suportadas pelo núcleo de operações criptográficas, que fornece serviços de criptografia baseada em identidade ao *middleware*. A integração desses componentes de segurança foi discutida em alguns cenários distintos, nos quais foram apresentadas algumas sugestões de configuração dos parâmetros de segurança do *middleware* para atender aos requisitos das aplicações.

O SEMAN fornece segurança ao sistema contra ataques de egoísmo, personificação, *Sybil*, e ataques bizantinos. Porém, outros tipos de ataques podem ser realizados nas MANETs que podem comprometer a eficácia do *middleware*. Como o SEMAN não foi avaliado considerando outros ataques, pode ser que o sistema não seja capaz de suportar outros tipos de ataques. Por exemplo, o sistema de gerenciamento de chaves baseado em identidade não foi avaliado considerando ataques do tipo homem no meio, e, portanto, não é possível afirmar que ele resistirá a esses tipos de ataques.

O núcleo de segurança utilizou a criptografia baseada em identidade, mas outras técnicas, que não necessitam a emissão de certificados, também poderiam ser utilizadas. Contudo, elas não foram estudadas neste trabalho. Por exemplo, a abordagem de criptografia de chave pública sem certificados, como apresentada em (GOYA; OKIDA; TERADA, 2010) poderia ter sido empregada. Com isso, o problema da custódia da chave privada seria eliminado sem a necessidade da criação de um PKG distribuído.

## 8.2 Publicações

Durante a realização dos estudos as seguintes publicações foram realizadas:

a. (MANNES et al., 2010): artigo publicado no **International Conference on Security and Cryptography (SECRYPT 2010)**, em julho de 2010, em parceria com Elisa Mannes, Michele Nogueira Lima e Aldri Luiz dos Santos;

b. (SILVA; SANTOS; ALBINI, 2010a): artigo publicado no **XXX Congresso da Sociedade Brasileira de Computação – Concurso de Teses e Dissertações**, em julho de 2010, em parceria com Aldri Luiz dos Santos e Luiz Carlos Pessoa Albini;

c. (SILVA; LIMA; ALBINI, 2010): artigo publicado no **International Telecommunications Symposium (ITS'10)**, em setembro de 2010, em parceria com Murilo Soares Lima e Luiz Carlos Pessoa Albini;

d. (SILVA; SANTOS; ALBINI, 2010b): artigo publicado no **X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SbSeg 2010) - Concurso de Teses e Dissertações**, em outubro de 2010, em parceria com Aldri Luiz dos Santos e Luiz Carlos Pessoa Albini;

e. (SILVA; e Silva; ALBINI, 2011): artigo publicado no **Mobile Networks and Management (MONAMI '11)**, em setembro de 2011, em parceria com Renan Fischer e Silva e Luiz Carlos Pessoa Albini;

f. (LIMA et al., 2011): artigo publicado no **IEEE Wireless Comunications**, em dezembro de 2011, em parceria com Michele Nogueira Lima, Aldri Luiz dos Santos e Luiz Carlos Pessoa Albini;

g. (SILVA et al., 2012): artigo publicado no **Journal of Selected Areas in Telecommunications (JSAT)**, em maio de 2012, em parceria com Michele Nogueira Lima, Aldri Luiz dos Santos e Luiz Carlos Pessoa Albini;

h. (SILVA; MISAGHI; ALBINI, 2012a): artigo publicado no **International Conference on Networked Digital Technologies (NDT '12)**, em abril de 2012, em parceria com Mehran Misaghi e Luiz Carlos Pessoa Albini;

i. (SILVA; MISAGHI; ALBINI, 2012b): artigo publicado no **Journal of Digital Information Management (JDIM)**, em outubro de 2012, em parceria com Mehran Misaghi e Luiz Carlos Pessoa Albini;

j. (SILVA; ALBINI; LIMA, 2013): artigo publicado na **Revista de Informática Teórica e Aplicada (RITA)**, em janeiro de 2013, em parceira com Murilo Soares Lima e Luiz Carlos Pessoa Albini;

k. (SILVA; ALBINI, 2013): artigo publicado no **IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)**, em outubro de 2013, em parceria com Luiz Carlos Pessoa Albini;

l. (SILVA; SILVA; ALBINI, 2014): artigo publicado no **Journal of Selected Areas in Telecommunications (JSAT)**, em janeiro de 2014, em parceria com Renan Fischer e Silva e Luiz Carlos Pessoa Albini; e

m. (SILVA; ALBINI, 2014): artigo publicado no **Journal of Network and Computer Applications**, em agosto de 2014, em parceria com Luiz Carlos Pessoa Albini.

## 8.3 Trabalhos futuros

Para aumentar a confiabilidade do sistema, novos serviços podem ser integrados ao SEMAN, e podem ser realizados em trabalhos futuros:

a. integrar com ferramentas de análise do ambiente externo, para auxiliar na configuração automática e dinâmica das políticas de segurança do *middleware*;

b. propor a integração com outras técnicas de criptografia de chave pública que não requerem certificados;

c. implementar e avaliar o *middleware* em cenários reais; e

d. elaborar um esquema de contabilização integrado ao gerenciamento de confiança para impedir que ataques de negação de serviço possam sobrecarregar o sistema com o envio de mensagens de controle desnecessárias.

# REFERÊNCIAS

ABOBA, B.; BLUNK, L.; VOLLBRECHT, J.; CARLSON, J. *RFC 3748: Extensible Authentication Protocol (EAP)*. 2004. Disponível em: <http://www.ietf.org/rfc/rfc3748-.txt>.

AGRAWAL, P.; GHOSH, R. K.; DAS, S. K. Cooperative black and gray hole attacks in mobile ad hoc networks. In: *Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication (ICUIMC '08)*. Nova Iorque, NY, EUA: ACM, 2008. p. 310–314. ISBN 978-1-59593-993-7.

AGRAWAL, S.; JAIN, S.; SHARMA, S. A survey of routing attacks and security measures in mobile ad-hoc networks. *Journal of Computing*, Journal of Computing Press, Nova Iorque, NY, EUA, v. 3, p. 41–48, jan. 2011. ISSN 2151-9617.

AIKEBAIER, A.; ENOKIDO, T.; TAKIZAWA, M. Trustworthy group formation algorithm based on decentralized trust management in distributed systems. In: BAROLLI, L.; TANIAR, D.; ENOKIDO, T.; RAHAYU, J. W.; TAKIZAWA, M. (Ed.). *Proceedings of the 15th International Conference on Network-Based Information Systems (NBiS '12)*. Melbourne, Austrália: IEEE Press, 2012. p. 58–65. ISBN 978-1-4673-2331-4.

AL-JAROODI, J.; JAWHAR, I.; AL-DHAHERI, A.; AL-ABDOULI, F.; MOHAMED, N. Security middleware approaches and issues for ubiquitous applications. *Computers and Mathematics with Applications*, Elsevier, Tarrytown, NY, EUA, v. 60, p. 187–197, jul. 2010. ISSN 0898-1221.

ALBERS, P.; CAMP, O.; PERCHER, J.-M.; JOUGA, B.; MÉ, L.; PUTTINI, R. S. Security in ad hoc networks: a general intrusion detection architecture enhancing trust based approaches. In: *Proceedings of the 1st International Workshop on Wireless Information Systems (WIS '02)*. Ciudad Real, Espanha: ICEIS Press, 2002. p. 1–12. ISBN 972-98816-0-X.

ALBINI, L. C. P.; CARUSO, A.; CHESSA, S.; MAESTRINI, P. Reliable routing in wireless ad hoc networks: The virtual routing protocol. *Journal of Network and Systems Management*, v. 14, p. 335–358, 2006.

ANJUM, F.; MOUCHTARIS, P. *Security for wireless ad hoc networks*. Hoboken, NJ, EUA: John Wiley & Sons, 2007. ISBN 978-0-471-75688-0.

ARRUFAT, M.; PARÍS, G.; LóPEZ, P. G. AGORA: An integrated approach for collaboration in MANETs. In: *Proceedings of the 1st International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (MOBILWARE '08)*. Bruxelas, Bélgica: ICST, 2008. p. 44:1–44:6. ISBN 978-1-59593-984-5.

ARRUFAT, M.; PARÍS, G.; LÓPEZ, P. G.; GÓMEZ-SKARMETA, A. F. SCOMET: Adapting collaborative working environments to the MANET scenario. In: *Proceedings of the 16th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE 2007)*. Paris, França: IEEE Computer Society, 2007. p. 106–110.

AUGOT, D.; BHASKAR, R.; ISSARNY, V.; SACCHETTI, D. An efficient group key agreement protocol for ad hoc networks. In: *Proceedings of the First International IEEE WoWMoM Workshop on Trust, Security and Privacy for Ubiquitous Computing (WOWMOM '05)*. Washington, DC, EUA: IEEE Computer Society, 2005. p. 576–580. ISBN 0-7695-2342-0-03.

AVVENUTI, M.; VECCHIO, A.; TURI, G. A cross-layer approach for publish/subscribe in mobile ad hoc networks. In: *Mobility Aware Technologies and Applications*. Berlim, Alemanha: Springer Berlin / Heidelberg, 2005, (Lecture Notes in Computer Science, v. 3744). p. 203–214. ISBN 0302-9743.

BAN, B. *JGroups Toolkit*. 2014. Disponível em `http://www.jgroups.org`. Acessado em março de 2014. Disponível em: <www.jgroups.org>.

BANAVAR, G.; CHANDRA, T. D.; STROM, R. E.; STURMAN, D. C. A case for message oriented middleware. In: *Proceedings of the 13th International Symposium on Distributed Computing (DISC '99)*. Bratislava, Eslováquia: Springer-Verlag, 1999. p. 1–18. ISBN 3-540-66531-5.

BANERJEE, U.; SWAMINATHAN, A. Taxonomy of attacks and attackers in MANETs. *International Journal of Research and Reviews in Computer Science (IJRRCS)*, Science Academy Publisher, Londres, Reino Unido, v. 2, abr. 2011. ISSN 2079-2557.

BELLAVISTA, P.; CORRADI, A.; MAGISTRETTI, E. Redman: A decentralized middleware solution for cooperative replication in dense manets. In: *Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW '05)*. Washington, DC, EUA: IEEE Computer Society, 2005. p. 158–162. ISBN 0-7695-2300-5.

BELLUR, U.; BONDRE, S. xSpace: a tuple space for XML & its application in orchestration of web services. In: *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC '06)*. Dijon, França: ACM, 2006. p. 766–772. ISBN 1-59593-108-2.

BERNSTEIN, P. A. Middleware: a model for distributed system services. *Communications of the ACM*, ACM, Nova Iorque, NY, EUA, v. 39, p. 86–98, 1996. ISSN 0001-0782.

BETH, T.; BORCHERDING, M.; KLEIN, B. Valuation of trust in open networks. In: *Proceedings of the 3rd European Symposium on Research in Computer Security (ESORICS '94))*. Londres, Reino Unido: Springer-Verlag, 1994. p. 3–18. ISBN 3-540-58618-0.

BISIGNANO, M.; CALVAGNA, A.; MODICA, G. D.; TOMARCHIO, O. ExPeerience: a JXTA middleware for mobile ad-hoc networks. In: *Proceedings of the 3rd International Conference on Peer-to-Peer Computing (P2P '03)*. Linköping, Suécia: IEEE Computer Society, 2003. p. 214–215. ISBN 0-7695-2023-5.

BISIGNANO, M.; CALVAGNA, A.; MODICA, G. D.; TOMARCHIO, O. Design and development of a JXTA middleware for mobile ad-hoc networks. In: *Proceedings of the International Conference on Parallel and Distributed Computing and Networks (PDCN '04)*. Innsbruck, Áustria: IASTED/ACTA Press, 2004. p. 177–182.

BISIGNANO, M.; MODICA, G. D.; TOMARCHIO, O. JMobiPeer: A middleware for mobile peer-to-peer computing in MANETs. In: *Proceedings of the 1st International Workshop on Mobility in Peer-to-Peer Systems (ICDCSW '05)*. Washington, DC, EUA: IEEE Computer Society, 2005. v. 8, p. 785–791. ISBN 0-7695-2328-5-08.

BLAZE, M.; FEIGENBAUM, J.; KEROMYTIS, A. D. The role of trust management in distributed systems security. In: *Secure Internet Programming*. Londres, Reino Unido: Springer, 1999. (Lecture Notes in Computer Science, v. 1603), p. 185–210. ISBN 3-540-66130-1.

BLAZE, M.; FEIGENBAUM, J.; LACY, J. Decentralized trust management. In: *Proceedings of the 1996 IEEE Symposium on Security and Privacy (SP '96)*. Oakland, CA, EUA: IEEE Computer Society, 1996. p. 164. ISBN 0-8186-7417-2.

BOHIO, M. J.; MIRI, A. Efficient identity-based security schemes for ad hoc network routing protocols. *Ad Hoc Networks*, Elsevier Science, Amsterdã, Países Baixos, v. 2, n. 3, p. 309–317, 2004.

BOLTON, G. E.; RAMI, Z. Anonymity versus punishment in ultimatum bargaining. *Games and Economic Behavior*, v. 10, n. 1, p. 95–121, jul. 1995.

BONEH, D.; FRANKLIN, M. K. Identity-based encryption from the weil pairing. In: *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '01)*. Londres, Reino Unido: Springer-Verlag, 2001. p. 213–229. ISBN 3-540-42456-3.

BOTTAZZI, D.; CORRADI, A.; MONTANARI, R. A context-aware group management middleware to support resource sharing in MANET environments. In: *Proceedings of the 6th International Conference on Mobile Data Management (MDM '05)*. Nova Iorque, NY, EUA: ACM, 2005. p. 147–151. ISBN 1-59593-041-8.

BOTTAZZI, D.; MONTANARI, R.; ROSSI, G. A self-organizing group management middleware for mobile ad-hoc networks. *Computer Communications*, Elsevier Science Publishers B. V., Amsterdã, Países Baixos, v. 31, n. 13, p. 3040–3048, ago. 2008. ISSN 0140-3664.

BOUKERCHE, A.; REN, Y. A security management scheme using a novel computational reputation model for wireless and mobile ad hoc networks. In: *Proceedings of the 5th ACM symposium on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks (PE-WASUN '08)*. Vancouver, Canadá: ACM, 2008. p. 88–95. ISBN 978-1-60558-236-8.

BOUKERCHE, A.; TURGUT, B.; AYDIN, N.; AHMAD, M. Z.; BöLöNI, L.; TURGUT, D. Routing protocols in ad hoc networks: A survey. *Computer Networks*, Elsevier North-Holland, Inc., Nova Iorque, NY, EUA, v. 55, p. 3032–3080, set. 2011. ISSN 1389-1286.

BRUSCHI, D.; ROSTI, E. Secure multicast in wireless networks of mobile hosts: protocols and issues. *Mobile Networks and Applications*, Kluwer Academic Publishers, Hingham, MA, EUA, v. 7, p. 503–511, 2002. ISSN 1383-469X.

BUCHEGGER, S.; BOUDEC, J.-Y. L. Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks. In: *Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*. Ilhas Canárias, Espanha: IEEE Computer Society, 2002. p. 403–410.

BUCHEGGER, S.; BOUDEC, J.-Y. L. Performance analysis of the confidant protocol. In: *Proceedings of the 3rd ACM International Symposium on Mobile ad hoc networking & computing (MobiHoc '02)*. Nova Iorque, NY, EUA: ACM, 2002. p. 226–236. ISBN 1-58113-501-7.

BUSKENS, V. *Social Networks and Trust*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2002. ISBN 1-4020-7010-1.

BUTTYÁN, L.; HUBAUX, J.-P. *Nuglets: a virtual currency to stimulate cooperation in self-organized ad hoc networks*. Lausanne, Suíça, 2001.

CABRI, G.; LEONARDI, L.; ZAMBONELLI, F. Mars: A programmable coordination architecture for mobile agents. *IEEE Internet Computing*, IEEE Educational Activities Department, Piscataway, NJ, EUA, v. 4, p. 26–35, jul. 2000. ISSN 1089-7801.

CAI, L.; PAN, J.; SHEN, X. S.; MARK, J. W. Promoting identity-based key management in wireless ad hoc networks. In: _____. *Wireless/Mobile Network Security*. Nova Iorque, NY, EUA: Springer US, 2007. (Springer Series on Signals and Communication Technology), p. 88–102.

CáRDENAS, A. A.; RADOSAVAC, S.; BARAS, J. S. Detection and prevention of mac layer misbehavior in ad hoc networks. In: *Proceedings of the 2nd ACM Workshop on Security of Ad hoc and Sensor Networks (SASN '04)*. Nova Iorque, NY, EUA: ACM, 2004. p. 17–22. ISBN 1-58113-972-1.

CHANDRA, P. *Bulletproof Wireless Security: GSM, UMTS, 802.11 and ad hoc Security*. Burlington, MA, EUA: Elsevier, 2005. (Communications Engeneering Series). ISBN 0-7506-7746-5.

CHANDRAKANT, N.; SHENOY, P. D.; VENUGOPAL, K. R.; PATNAIK, L. M. Middleware services for security in scalable and non-scalable heterogeneous nodes of MANETs. *International Journal of Future Generation Communication and Networking*, Science & Engineering Research Support Society, Tasmania, Austrália, n. 2, jun. 2011. ISSN 2233-7857.

CHANDRAKANT, N.; SHENOY, P. D.; VENUGOPAL, K. R.; PATNAIK, L. M. Restricting the admission of selfish or malicious nodes into the network by using efficient security services in middleware for MANETs. In: *Proceedings of the 2011 International Conference on Communication, Computing & Security (ICCCS '11)*. Nova Iorque, NY, EUA: ACM, 2011. p. 489–492. ISBN 978-1-4503-0464-1.

CHANG, B.-J.; KUO, S.-L.; LIANG, Y.-H.; WANG, D.-Y. Markov chain-based trust model for analyzing trust value in distributed multicasting mobile ad hoc networks. IEEE Computer Society, Washington, DC, EUA, v. 59, p. 1846–1863, 2009. ISSN 0018-9545.

CHEN, Z.; GUO, S.; ZHENG, K.; YANG, Y. Modeling of man-in-the-middle attack in the wireless networks. In: *Proceedings of the 2007 International Conference on Wireless Communications, Networking and Mobile Computing (WiCom '07)*. Xangai, China: IEEE Communications Society, 2007. p. 2255 –2258. ISBN 978-1-4244-1311-9.

CHIEN, H.-Y.; LIN, R.-Y. Improved ID-based security framework for ad hoc network. *Ad Hoc Networks*, Elsevier Science, Amsterdã, Países Baixos, v. 6, n. 1, p. 47–60, 2008. ISSN 1570-8705.

CHLAMTAC, I.; CONTI, M.; LIU, J. J.-N. Mobile ad hoc networking: imperatives and challenges. *Ad Hoc Networks*, Elsevier Science, Amsterdã, Países Baixos, v. 1, n. 1, p. 13–64, 2003. ISSN 1570-8705.

CHO, J.-H.; SWAMI, A.; CHEN, I.-R. A survey on trust management for mobile ad hoc networks. *Communications Surveys Tutorials, IEEE*, v. 13, n. abr., p. 562–583, out. 2011. ISSN 1553-877X.

CHUNLIN, L.; ZHENGDING, L.; LAYUAN, L.; SHUZHI, Z. A mobile agent platform based on tuple space coordination. *Advances in Engineering Software*, v. 33, n. 4, p. 215–225, 2002. ISSN 0965-9978.

CLAUSEN, T.; JACQUET, P. *RFC3626 - Optimized Link State Protocol (OLSR)*. 2003. Disponível em: <http://www.ietf.org/rfc/rfc3626.txt>.

CONTI, M.; MASELLI, G.; TURI, G.; GIORDANO, S. Cross-layering in mobile ad hoc network design. *IEEE Computer*, IEEE Computer Society Press, Los Alamitos, CA, EUA, v. 37, p. 48–51, 2004. ISSN 0018-9162.

COSTAGLIOLA, N.; LóPEZ, P. G.; OLIVIERO, F.; ROMANO, S. P. Energy- and delay-efficient routing in mobile ad hoc networks. *Mobile Networks and Applications*, Springer US, v. 17, p. 281–297, 2012. ISSN 1383-469X.

COURAND, O.; DROEGEHORN, O.; DAVID, K.; NURMI, P.; FLOREEN, P.; KERNCHEN, R.; HOLTMANNS, S.; CAMPADELLO, S.; KANTER, T.; MARTIN, M.; EIJK, R. van; GUARNERI, R. Context aware group management in mobile environments. In: *Proceedings of the 14th IST Mobile and Wireless Communications Summit 2005*. Dresden, Alemanha: Nokia Research Center, 2005.

CROSON, R.; BUCHAN, N. Gender and culture: International experimental evidence from trust games. *The American Economic Review*, JSTOR, v. 89, n. 2, p. 386–391, maio 1999.

DAEMEN, J.; RIJMEN, V. *The Design of Rijndael*. Secaucus, NJ, EUA: Springer-Verlag New York, Inc., 2002. ISBN 3540425802.

DAI, H.; JIA, Z.; QIN, Z. Trust evaluation and dynamic routing decision based on fuzzy theory for MANETs. *JSW – Journal of Software*, Academy Publisher, v. 4, n. 10, p. 1091–1101, 2009.

DAVIES, N.; FRIDAY, A.; WADE, S. P.; BLAIR, G. S. L2imbo: a distributed systems platform for mobile computing. *Mobile Networks and Applications*, Kluwer Academic Publishers, Hingham, MA, EUA, v. 3, p. 143–156, ago. 1998. ISSN 1383-469X.

DAZA, V.; MORILLO, P.; RàFOLS, C. On dynamic distribution of private keys over manets. *Electronic Notes in Theoretical Computer Science (ENTCS)*, Elsevier Science Publishers B. V., v. 171, n. 1, p. 33–41, 2007. ISSN 1571-0661.

DEFRAWY, K. E. *Security and Privacy in Location-based Mobile Ad-hoc Networks*. Tese (Doutorado) — California State University, Long Beach, CA, EUA, 2010.

DELLAROCAS, C. Mechanisms for coping with unfair ratings and discriminatory behavior in online reputation reporting systems. In: *Proceedings of the 21th International Conference on Information Systems (ICIS '00)*. Atlanta, GA, EUA: Association for Information Systems, 2000. p. 520–525. ISBN ICIS2000-X.

DEMEURE, I.; PAROUX, G.; HERNANDO-URETA, J.; KHAKPOUR, A. R.; NOWALCZYK, J. An energy-aware middleware for collaboration on small scale MANETS. In: *Proceedings of the Autonomous and Spontaneous Networks Symposium (ASNS '08)*. Paris, França: Institut TELECOM, 2008.

DENG, H.; LI, W.; AGRAWAL, D. P. Routing security in wireless ad hoc networks. *IEEE Communications Magazine*, v. 40, n. 10, p. 70–75, out. 2002.

DENG, H.; MUKHERJEE, A.; AGRAWAL, D. P. Threshold and identity-based key management and authentication for wireless ad hoc networks. In: *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)*. Washington, DC, EUA: IEEE Computer Society, 2004. v. 2, p. 107. ISBN 0-7695-2108-8.

DENKO, M. K. A markov model-based location prediction scheme for mobile ad-hoc networks. In: *Proceedings of the International Conference on Wireless Networks (ICWN '04)*. Las Vegas, NE, EUA: CSREA Press, 2004. p. 179–184. ISBN 1-932415-38-6.

DENKO, M. K.; SHAKSHUKI, E.; MALIK, H. Enhanced cross-layer based middleware for mobile ad hoc networks. *Journal of Network and Computer Applications*, Elsevier, Londres, Reino Unido, v. 32, n. 2, p. 490–499, 2009. ISSN 1084-8045.

DESILVA, S.; BOPPANA, R. Mitigating malicious control packet floods in ad hoc networks. In: *Procedding of the 2005 IEEE Wireless Communications and Networking Conference (WCNC '05)*. Nova Orleans, LA, EUA: [s.n.], 2005. v. 4, p. 2112–2117. ISSN 1525-3511.

DJENOURI, D.; KHELLADI, L.; BADACHE, N. A survey of security issues in mobile ad hoc and sensor networks. *IEEE Surveys and Tutorials*, IEEE Communications Society, Los Alamitos, CA, EUA, v. 7, n. 4, p. 2–28, 2005. ISSN 1553-877X.

DOUCEUR, J. R. The Sybil attack. In: *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS '01)*. Londres, Reino Unido: Springer-Verlag, 2001. p. 251–260. ISBN 3-540-44179-4.

EARLE, A. E. *Wireless Security Handbook*. Boca Raton, FL, EUA: Auerback publications, 2006. ISBN 978-0-8493-3378-1.

FOK, C.-L.; ROMAN, G.-C.; HACKMANN, G. A lightweight coordination middleware for mobile computing. In: *Proceedings of the 6th International Conference of Coordination Models and Languages (COORDINATION '04)*. Pisa, Itália: Springer, 2004. (Lecture Notes in Computer Science, v. 2949), p. 135–151. ISBN 3-540-21044-X.

FREY, D.; ROMAN, G.-C. Context-aware publish subscribe in mobile ad hoc networks. In: *Proceedings of the 9th International Conference on Coordination Models and Languages (COORDINATION '07)*. Paphos, Chipre: Springer, 2007. (Lecture Notes in Computer Science, v. 4467), p. 37–55. ISBN 978-3-540-72793-4.

GALE, J.; BINMORE, K. G.; SAMUELSON, L. Learning to be imperfect: The ultimatum game. *Games and Economic Behavior*, v. 8, n. 1, p. 56–90, 1995. ISSN 0899-8256.

GEIHS, K. Middleware challenges ahead. *Computer*, IEEE Computer Society Press, Los Alamitos, CA, EUA, v. 34, p. 24–31, jun. 2001. ISSN 0018-9162.

GELERNTER, D. Generative communication in Linda. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, ACM, Nova Iorque, NY, EUA, v. 7, n. 1, p. 80–112, 1985. ISSN 0164-0925.

GHOSH, A.; LI, S. wei; CHIANG, C. J.; CHADHA, R.; MOELTNER, K.; ALI, S.; KUMAR, Y.; BAUER, R. QoS-aware Adaptive Middleware (QAM) for tactical MANET applications. In: *Proceedings of the 2010 Military Communications Conference (MILCOM '10)*. San Jose, CA, EUA: IEEE Communications Society, 2010. p. 178–183. ISSN 978-1-4244-8179-8.

GHOSH, T.; PISSINOU, N.; MAKKI, K. Towards designing a trusted routing solution in mobile ad hoc networks. *Mobile Networks and Applications*, Kluwer Academic Publishers, Hingham, MA, EUA, v. 10, n. 6, p. 985–995, 2005. ISSN 1383-469X.

GOLBECK, J. Computing with trust: Definition, properties, and algorithms. In: *Proceedings of the 1st International Conference on Security and Privacy in Communications Networks and the Workshops (SecureComm '06)*. Baltimore, MD, EUA: IEEE Press, 2006. p. 1–7.

GORLATOVA, M. A.; MASON, P. C.; WANG, M.; LAMONT, L.; LISCANO, R. Detecting wormhole attacks in mobile ad hoc networks through protocol breaking and packet timing analysis. In: *Proceedings of the 2006 IEEE Conference on Military Communications (MILCOM '06)*. Piscataway, NJ, EUA: IEEE Press, 2006. p. 1068–1074. ISBN 1-4244-0618-8.

GOSPIC, K.; MOHLIN, E.; FRANSSON, P.; PETROVIC, P.; JOHANNESSON, M.; INGVAR, M. Limbic justice - amygdala involvement in immediate rejection in the ultimatum game. *PLoS Biology*, Public Library of Science, v. 9, n. 5, p. 1–8, maio 2011.

GOYA, D.; OKIDA, C.; TERADA, R. A two-party certificateless authenticated key agreement protocol. In: *Anais do X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SbSeg 2010)*. Fortaleza, CE, Brasil: SBC, 2010. p. 433–446.

GUTH, W.; SCHMITTBERGER, R.; SCHWARZE, B. An experimental analysis of ultimatum bargaining. *Journal of Economic Behavior & Organization*, v. 3, n. 4, p. 367–388, 1982. ISSN 0167-2681.

GöRGEN, D.; FREY, H.; LEHNERT, J. K.; STURM, P. SELMA: A middleware platform for self-organzing distributed applications in mobile multihop ad-hoc networks. In: *Proceedings of the 2004 Communication Networks and Distributed Systems Modeling*

*and Simulation (CNDS' 04)*. San Diego, CA, EUA: Society for Modeling and Simulation International, 2004.

HAAS, Z. J.; PEARLMAN, M. R. ZRP: a hybrid framework for routing in ad hoc networks. *Ad hoc networking*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, EUA, p. 221–253, 2001.

HADIM, S.; AL-JAROODI, J.; MOHAMED, N. Middleware issues and approaches for mobile ad hoc networks. In: *Proceedings of the 3rd IEEE Consumer Communications and Networking Conference (CCNC '06)*. Las Vegas, NV, EUA: IEEE, 2006. v. 1, p. 431–436.

HADIM, S.; AL-JAROODI, J.; MOHAMED, N. Trends in middleware for mobile ad hoc networks. *Journal of Communications*, v. 1, n. 4, 2006.

HAPNER, M.; BURRIDGE, R.; SHARMA, R.; FIALLI, J.; STOUT, K. *Java Message Service Specification - Version 1.1*. Palo Alto, CA, EUA, 2002. Disponível em: <http://java.sun.com/products/jms/>.

HE, Q.; WU, D.; KHOSLA, P. SORI: A secure and objective reputation-based incentive scheme for ad-hoc networks. In: *Proceedings of the 2004 IEEE Wireless Communications and Networking Conference (WCNC '04)*. Atlanta, GA, EUA: IEEE Communications Society, 2004. p. 825–830.

HERRMANN, K.; MüHL, G.; JAEGER, M. A. MESHMdl event spaces - A coordination middleware for self-organizing applications in ad hoc networks. *Pervasive and Mobile Computing*, Elsevier Science Publishers B. V., Amsterdã, Países Baixos, v. 3, n. 4, p. 467–487, 2007. ISSN 1574-1192.

HOEPER, K.; GONG, G. *Bootstrapping Security in Mobile Ad Hoc Networks Using Identity-Based Schemes with Key Revocation*. Waterloo, ON, Canadá, 2006.

HOEPER, K.; GONG, G. Key revocation for identity-based schemes in mobile ad hoc networks. In: *Proceedings of the 5th International Conference on Ad-Hoc, Mobile, and Wireless Networks (ADHOC-NOW '06)*. Ottawa, Canadá: Springer-Verlag, 2006. p. 224–237. ISBN 3-540-37246-6, 978-3-540-37246-2.

HU, Y.-C.; PERRIG, A. A survey of secure wireless ad hoc routing. *IEEE Security and Privacy*, IEEE Educational Activities Department, Piscataway, NJ, EUA, v. 2, n. 3, p. 28–39, 2004. ISSN 1540-7993.

HU, Y.-C.; PERRIG, A.; JOHNSON, D. Wormhole attacks in wireless networks. *IEEE Journal on Selected Areas in Communications*, v. 24, p. 370–380, 2006. ISSN 0733-8716.

HU, Y. C.; PERRIG, A.; JOHNSON, D. B. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In: *Proceedings of 22nd Annual Joint Conferecente of the IEEE Computer and Communications Societies (INFOCOM '03)*. San Francisco, CA, EUA: [s.n.], 2003. v. 3, p. 1976–1986.

HU, Y.-C.; PERRIG, A.; JOHNSON, D. B. Rushing attacks and defense in wireless ad hoc network routing protocols. In: *Proceedings of the 2nd ACM Workshop on Wireless Security (WiSe '03)*. Nova Iorque, NY, EUA: ACM, 2003. p. 30–40. ISBN 1-58113-769-9.

HU, Y.-C.; PERRIG, A.; JOHNSON, D. B. Ariadne: a secure on-demand routing protocol for ad hoc networks. *Wireless Networks*, Kluwer Academic Publishers, Hingham, MA, EUA, v. 11, n. 1-2, p. 21–38, 2005. ISSN 1022-0038.

HUR, J.; PARK, C.; HWANG, S. O. Privacy-preserving identity-based broadcast encryption. *Information Fusion*, Elsevier, Amsterdã, Países Baixos, v. 13, n. 4, p. 296–303, out. 2012. ISSN 1566-2535.

IEEE. *IEEE Standard for Wireless LAN-Medium Access Control and Physical Layer Specification*. Nova Iorque, NY, EUA, 1999. IEEE Strandard 802.11-1999.

IEEE. *IEEE Standards for Local and Metropolitan Area Networks: Port based Network Access Control*. Nova Iorque, NY, 2004.

J2ME. *Java 2 Micro Edition (J2ME)*. 2014. Disponível em `http://java.sun.com/j2me/`. Acessado em março de 2014. Disponível em: <http://java.sun.com/j2me/>.

JIANG, T.; BARAS, J. S. Ant-based adaptive trust evidence distribution in manet. In: *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCSW'04)*. Tóquio, Japão: IEEE Computer Society, 2004. p. 588–593. ISBN 0-7695-2087-1.

JOHNSON, D. B.; MALTZ, D. A. Dynamic source routing in ad hoc wireless networks. In: *Mobile Computing*. Hingham, MA, EUA: Kluwer Academic Publishers, 1996. v. 353, p. 153–181.

JUNG, B. E. An efficient group key agreement protocol. *IEEE Communications Letters*, v. 10, n. 2, p. 106–107, fev. 2006. ISSN 1089-7798.

JXTA. *The JXTA Project*. 2014. Disponível em `http://www.jxta.org/`. Acessado em março de 2014. Disponível em: <http://www.jxta.org/>.

KANNHAVONG, B.; NAKAYAMA, H.; NEMOTO, Y.; KATO, N.; JAMALIPOUR, A. A survey of routing attacks in mobile ad hoc networks. *IEEE Wireless Communications*, v. 14, p. 85–91, 2007. ISSN 1536-1284.

KATE, A.; GOLDBERG, I. Distributed private-key generators for identity-based cryptography. In: *Proceedings of the 7th International Conference on Security and Cryptography for Networks (SCN'10)*. Berlin, Alemanha: Springer-Verlag, 2010. p. 436–453. ISBN 3-642-15316-X, 978-3-642-15316-7.

KHALILI, A.; KATZ, J.; ARBAUGH, W. A. Toward secure key distribution in truly ad-hoc networks. In: *Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT '03)*. Washington, DC, EUA: IEEE Computer Society, 2003. p. 342. ISBN 0-7695-1873-7.

KIM, Y.; MAZZOCCHI, D.; TSUDIK, G. Admission control in peer groups. In: *Proceedings of the 2nd IEEE International Symposium on Network Computing and Applications (NCA '03)*. Washington, DC, EUA: IEEE Computer Society, 2003. p. 131. ISBN 0-7695-1938-5.

KONG, J.; ZERFOS, P.; LUO, H.; LU, S.; ZHANG, L. Providing robust and ubiquitous security support for mobile ad hoc networks. In: *Proceedings of the 9th International Conference on Network Protocols (ICNP '01)*. Washington, DC, EUA: IEEE Computer Society, 2001. p. 251.

KORTUEM, G. Proem: a middleware platform for mobile peer-to-peer computing. *SIGMOBILE Mobile Computing Communications Review*, ACM, Nova Iorque, NY, EUA, v. 6, p. 62–64, 2002. ISSN 1559-1662.

KUMAR, A.; GUPTA, D. P.; VERMA, P. K.; LAMBA, D. V. Concept of middleware services in mobile ad-hoc networks. *International Journal of Computer Applications*, Foundation of Computer Science, v. 2, n. 8, p. 33–36, jun. 2010.

KUROSAWA, S.; NAKAYAMA, H.; KATO, N.; JAMALIPOUR, A.; NEMOTO, Y. Detecting blackhole attack on AODV-based mobile ad hoc networks by dynamic learning method. *International Journal of Network Security*, v. 5, p. 338–346, 2007.

KYASANUR, P.; VAIDYA, N. H. Detection and handling of mac layer misbehavior in wireless networks. In: *Proceedings of the 2003 IEEE/IFIP International Conference on Dependable Systems and Networks (DSN '03)*. Los Alamitos, CA, EUA: IEEE Computer Society, 2003. p. 173–182. ISBN 0-7695-1952-0.

LAHYANI, I.; RODRIGUEZ, I. B.; JMAIEL, M.; CHASSOT, C. Towards self healing publish/subscribe system on manet. In: *Proceedings of the IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*. Los Alamitos, CA, EUA: IEEE Computer Society, 2012. p. 385–390. ISSN 1524-4547.

LEWIS, J. D.; WEIGERT, A. Trust as a social reality. *Social Forces*, University of North Carolina Press, v. 63, n. 4, p. 967–985, 1985. ISSN 00377732.

LI, X.; SLAY, J.; YU, S. Evaluating trust in mobile ad hoc networks. In: *Proceedings of the 2005 Workshop of International Conference on Computational Intelligence and Security (CIS '05)*. Xi'an, China: Springer, 2005. ISSN 0302-9743.

LIMA, M. N.; PUJOLLE, G.; SILVA, E. da; SANTOS, A. L. dos; ALBINI, L. C. P. Survivable keying for wireless ad hoc networks. In: *Proceedings of the 2009 IFIP/IEEE International Symposium on Integrated Network Management (IM '09)*. Nova Iorque, NY, EUA: IEEE Communications Society, 2009. p. 606–613.

LIMA, M. N.; SILVA, E. da; SANTOS, A. L. dos; ALBINI, L. C. P. Surviving key management on wanets. *IEEE Wireless Communications Magazine*, IEEE Communications Society, Nova Iorque, NY, EUA, v. 18, p. 82–88, dez. 2011. ISSN 1536-1284.

LIU, C.; SHU, Y.; LI, M.; YANG, O. W. W. A new mechanism to detect selfish behavior in IEEE 802.11 ad hoc networks. In: *Proceedings of the 2009 IEEE International Conference on Communications (ICC '09)*. Piscataway, NJ, EUA: IEEE, 2009. p. 4918–4922. ISBN 978-1-4244-3434-3.

LIU, J.; SACCHETTI, D.; SAILHAN, F.; ISSARNY, V. Group management for mobile ad hoc networks: Design, implementation and experiment. In: *Proceedings of the 6th*

*International Conference on Mobile Data Management (MDM '05)*. Nova Iorque, NY, EUA: ACM, 2005. p. 192–199. ISBN 1-59593-041-8.

LIU, W. Securing mobile ad hoc networks with certificateless public keys. *IEEE Transactions on Dependable Secure Computing*, IEEE Computer Society Press, Los Alamitos, CA, EUA, v. 3, n. 4, p. 386–399, 2006. ISSN 1545-5971.

LOPEZ, P. G.; TINEDO, R. G.; PALAU, M. E.; MESSEGUER, R. Topology-aware group communication middleware for MANETs. In: *Proceedings of the 4th International ICST Conference on Communication System software and middleware (COMSWARE '09)*. Dublin, Irlanda: ACM, 2009. p. 1–10. ISBN 978-1-60558-353-2.

LUO, H.; KONG, J.; ZERFOS, P.; LU, S.; ZHANG, L. URSA: ubiquitous and robust access control for mobile ad hoc networks. *IEEE/ACM Transactions on Networking*, IEEE Communications Society, Piscataway, NJ, EUA, v. 12, n. 6, p. 1049–1063, 2004. ISSN 1063-6692.

MAMEI, M.; ZAMBONELLI, F.; LEONARDI, L. Tuples on the air: A middleware for context-aware computing in dynamic networks. In: *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS '03)*. Providence, RI, EUA: IEEE Computer Society, 2003. p. 342–350. ISBN 0-7695-1921-0.

MANNES, E.; SILVA, E. da; LIMA, M. N.; SANTOS, A. L. dos. Implications of misbehaving attacks on probalistic quorum system for manets. In: *Proceedings of the International Conference on Security and Cryptography (SECRYPT '10)*. Atenas, Grécia: INSTCC Press, 2010. p. 189–195.

MARCHETTI, A.; CASTELLI, I.; HARLé, K. M.; SANFEY, A. G. Expectations and outcome: The role of proposer features in the ultimatum game. *Journal of Economic Psychology*, Elsevier B.V., v. 32, n. 3, p. 446–449, jun. 2011. ISSN 0167-4870.

MARIAS, G. F.; GEORGIADIS, P.; FLITZANIS, D.; MANDALAS, K. Cooperation enforcement schemes for MANETs: a survey. *Wireless Communication Mobile Computing*, John Wiley and Sons, Chichester, Reino Unido, v. 6, n. 3, p. 319–332, 2006. ISSN 1530-8669.

MARTI, S.; GIULI, T. J.; LAI, K.; BAKER, M. Mitigating misbehavior in mobile ad hoc networks. In: *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom '00)*. Nova Iorque, NY, EUA: ACM, 2000. p. 255–265. ISBN 1-58113-197-6.

MARUTA, T.; OKADA, A. Dynamic group formation in the repeated Prisoner's dilemma. *Games and Economic Behavior*, Elsevier, Amsterdã, Países Baixos, v. 74, p. 269–284, 2012. ISSN 0899-8256.

MASCOLO, C.; CAPRA, L.; EMMERICH, W. Mobile computing middleware. In: _____. Nova Iorque, NY, EUA: Springer-Verlag New York, Inc., 2002. p. 20–58. ISBN 3-540-00165-4.

MASCOLO, C.; CAPRA, L.; ZACHARIADIS, S.; EMMERICH, W. XMIDDLE: A Data-Sharing Middleware for Mobile Computing. *Wireless Personal Communication*, Kluwer, p. 77–103, 2002.

MEIER, R.; CAHILL, V. STEAM: Event-based middleware for wireless ad hoc networks. In: *Proceedings of the 22th International Conference on Distributed Computing Systems Workshops (ICDCS '02)*. Viena, Áustria: IEEE Computer Society, 2002. p. 639–644. ISBN 0-7695-1588-6.

MEIER, R.; CAHILL, V. Distributed Applications and Interoperable Systems. In: _____. Berlim, Alemanha: Springer, 2003. (Lecture Notes in Computer Science, v. 2893/2003), cap. Exploiting Proximity in Event-Based Middleware for Collaborative Mobile Applications, p. 285–296. ISBN 978-3-540-20529-6.

MEJIA, M.; nA, N. P.; nOZ, J. L. M.; ESPARZA, O.; ALZATE, M. a. A game theoretic trust model for on-line distributed evolution of cooperation in manets. *Journal of Network and Computer Applications*, v. 34, n. 1, p. 39–51, jan. 2011.

MENEZES, A. J.; OORSCHOT, P. C. V.; VANSTONE, S. A. *Handbook of Applied Cryptography*. Danvers, MA, EUA: CRC Press, 1996.

MERWE, J. van der; DAWOUD, D.; MCDONALD, S. A survey on peer-to-peer key management for mobile ad hoc networks. *ACM Computing Survey*, ACM Press, Nova Iorque, NY, EUA, v. 39, n. 1, p. 1, 2007. ISSN 0360-0300.

MICHIARDI, P.; MOLVA, R. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In: *Proceedings of the IFIP TC6/TC11 6th Joint Working Conference on Communications and Multimedia Security*. Deventer, Amsterdã, Países Baixos: Kluwer, B.V., 2002. p. 107–121. ISBN 1-4020-7206-6.

MICHIARDI, P.; MOLVA, R. Ad hoc networks security. *ST Journal of System Research*, STMicroelectronics, Genebra, Suíça, v. 4, n. 1, mar. 2003.

MICKENS, J. W.; NOBLE, B. D. Modeling epidemic spreading in mobile environments. In: *Proceedings of the 4th ACM Workshop on Wireless Security (WiSe '05)*. Nova Iorque, NY, EUA: ACM, 2005. p. 77–86. ISBN 1-59593-142-2.

MOLVA, R.; MICHIARDI, P. Security in ad hoc networks. In: *Proceeding of 8th IFIP International Conference on Personal Wireless Communications (PWC '03)*. Veneza, Itália: IFIP, 2003. Also published as LNCS Volume 2775.

MURPHY, A. L.; PICCO, G. P.; ROMAN, G.-C. LIME: A middleware for physical and logical mobility. In: *Proceedings of the the 21st International Conference on Distributed Computing Systems (ICDCS '01)*. Phoenix, AZ, EUA: IEEE Computer Society, 2001. p. 524–533.

MURPHY, A. L.; PICCO, G. P.; ROMAN, G.-C. LIME: A coordination model and middleware supporting mobility of hosts and agents. *ACM Transactions on Software Engineering and Methodology*, ACM, Nova Iorque, NY, EUA, v. 15, n. 3, p. 279–328, 2006. ISSN 1049-331X.

MUSOLESI, M.; MASCOLO, C.; HAILES, S. Emma: Epidemic messaging middleware for ad hoc networks. *Personal and Ubiquitous Computing*, Springer-Verlag, Londres, Reino Unido, v. 10, n. 1, p. 28–36, dez. 2005. ISSN 1617-4909.

NEWSOME, J.; SHI, E.; SONG, D.; PERRIG, A. The Sybil attack in sensor networks: analysis & defenses. In: *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN '04)*. Nova Iorque, NY, EUA: ACM, 2004. p. 259–268. ISBN 1-58113-846-6.

NICHELE, C. R. *Modelo de confiança para redes ad hoc baseado em teoria de jogos*. Dissertação (Dissertação) — Universidade Federal do Paraná (UFPR), Curitiba, PR, 2012.

NOWAK, M. A.; PAGE, K. M.; SIGMUND, K. Fairness versus reason in the ultimatum game. *Science*, v. 289, n. 5485, p. 1773–1775, 2000.

PAGE, E. S. Continuous inspection schemes. *Biometrika*, v. 41, n. 1/2, p. 100–115, jun. 1954.

PAN, J.; CAI, L.; SHEN, X.; MARK, J. W. Identity-based secure collaboration in wireless ad hoc networks. *Computer Networks*, Elsevier Science, v. 51, n. 3, p. 853–865, 2007.

PAPADIMITRATOS, P.; HAAS, Z. J. Securing mobile ad hoc networks. In: _____. *Mobile Computing Handbook*. Boca Raton, FL, EUA: CRC Press - Auerbach Publications, 2005. cap. 21, p. 457–481.

PARK, B.-N.; LEE, W. ISMANET: A secure routing protocol using identity-based signcryption scheme for mobile ad-hoc networks. *IEICE Transactions on Communications*, The Institute of Electronics, Information and Communication Engineers, v. 88, n. 6, p. 2548–2556, 2005. ISSN 09168516.

PARK, V.; CORSON, M. A highly adaptive distributed routing algorithm for mobile wireless networks. In: *Proceedings of the 16rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '97)*. Kobe, Japão: IEEE Communications Society, 1997. p. 1405–1413.

PERKINS, C.; BELDING-ROYER, E. Ad-hoc on-demand distance vector routing. In: *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*. Los Alamitos, CA, EUA: IEEE Computer Society, 1999. p. 90–100.

PERKINS, C.; BELDING-ROYER, E.; DAS, S. *RFC 3561: Ad hoc On-Demand Distance Vector (AODV) Routing*. 2003. Disponível em: <http://www.ietf.org/rfc/rfc3748.txt>.

PERKINS, C.; BHAGWAT, P. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In: *Proceedings of the ACM Conference on Communications Architectures, Protocols and Applications (SIGCOMM'94)*. Nova Iorque, NY, EUA: ACM Press., 1994. p. 234–244.

QIAN, L.; SONG, N.; LI, X. Detection of wormhole attacks in multi-path routed wireless ad hoc networks: a statistical analysis approach. *Journal of Network and Computer Applications - Special issue: Network and information security: A computational intelligence approach*, Academic Press Ltd., Londres, Reino Unido, v. 30, p. 308–330, 2007. ISSN 1084-8045.

QIAN, W.; CHEN, H.; LI, Z.; JIA, C. On a practical distributed key generation scheme based on bivariate polynomials. In: *Proceedings of the 7th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM '11)*. [S.l.: s.n.], 2011. p. 1–4. ISSN 2161-9646.

QIAN, W.; JIA, H. C. andCheng. A new-member-joining protocol using bivariate polynomials based distributed key generations. *WAP Conference Series: Information Science and Technology*, World Academic Publishing, Hong Kong, China, 2012. ISSN 2227-8060.

RACHEDI, A.; BENSLIMANE, A.; OTROK, H.; MOHAMMED, N.; DEBBABI, M. A secure mechanism design-based and game theoretical model for manets. *Mobile Networks and Applications*, Springer-Verlag New York, Inc., Secaucus, NJ, EUA, v. 15, n. 2, p. 191–204, abr. 2010. ISSN 1383-469X.

RAJ, P. N.; SWADAS, P. B. DPRAODV: A dyanamic learning system against blackhole attack in AODV-based manet. *International Journal of Computer Science Issues- IJCSI*, v. 2, 2009. ISSN 1694-0814.

RAPPAPORT, T. *Wireless Communications: Principles and Practice*. 2. ed. Upper Saddle River, NJ, EUA: Prentice Hall PTR, 2001. ISBN 0130422320.

RIGNEY, C.; WILLENS, S.; RUBENS, A.; SIMPSON, W. *RFC 2865: Remote Authentication Dial In User Service (RADIUS)*. 2000. Disponível em: <http://www.ietf-.org/rfc/rfc2865.txt>.

RIPEANU, M.; FOSTER, I.; IAMNITCHI, A. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system. *IEEE Internet Computing Journal*, IEEE Computer Society, Washington, DC, EUA, v. 6, p. 2002, 2002.

ROMAN, G.-C.; HANDOREAN, R.; SEN, R. Tuple space coordination across space and time. In: CIANCARINI, P.; WIKLICKY, H. (Ed.). *Coordination Models and Languages*. Berlim, Alemanha: Springer Berlin / Heidelberg, 2006, (Lecture Notes in Computer Science, v. 4038). p. 266–280.

SáNCHEZ-ARTIGAS, M.; ARRUFAT, M.; PARíS, G.; LóPEZ, P. G. SCOMET: A Middleware for Collaborative Working Environments in MANETs. *Ubiquitous Computing and Communications Journal*, v. 3, mar. 2008. ISSN 1992-8424.

SANFEY, A. G.; RILLING, J. K.; ARONSON, J. A.; NYSTROM, L. E.; COHEN, J. D. The neural basis of economic decision-making in the ultimatum game. *Science*, v. 300, n. 5626, p. 1755–1758, 2003.

SANZGIRI, K.; DAHILL, B.; LEVINE, B. N.; SHIELDS, C.; BELDING-ROYER, E. M. A secure routing protocol for ad hoc networks. In: *Proceedings of the 10th IEEE International Conference on Network Protocols (ICNP '02)*. Washington, DC, EUA: IEEE Computer Society, 2002. p. 78–89. ISBN 0-7695-1856-7.

SAXENA, N.; TSUDIK, G.; YI, J. Identity-based access control for ad hoc groups. In: PARK, C.-s.; CHEE, S. (Ed.). *Information Security and Cryptology – ICISC 2004*. Berlim, Alemanha: Springer Berlin Heidelberg, 2005, (Lecture Notes in Computer Science, v. 3506). p. 362–379. ISBN 978-3-540-26226-8.

SHAMIR, A. How to share a secret. *Communications of the ACM*, ACM, Nova Iorque, NY, EUA, v. 22, n. 11, p. 612–613, 1979. ISSN 0001-0782.

SHAMIR, A. Identity-based cryptosystems and signature schemes. In: *Proceedings of Advances in Cryptology (CRYPTO 84)*. Nova Iorque, NY, EUA: Springer-Verlag New York, Inc., 1985. p. 47–53. ISBN 0-387-15658-5.

SHIFERAW, A. N.; LAJMI, S.; SCUTURICI, V.-M.; BRUNIE, L. Pasmi: self-adaptive photo annotation and sharing middleware of mobile ad-hoc networks. In: *Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops 2010)*. Mannheim, Alemanha: IEEE, 2010. p. 135–140.

SHIREY, R. *RFC 2828: Internet security glossary*. Marina del Rey, CA, EUA, 2000. Disponível em: <http://www.ietf.org/rfc/rfc2828.txt>.

SILVA, E. da; ALBINI, L. C. P. Towards a fully self-organized identity-based key management system for MANETs. In: *Proceedings of the 9th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob '13)*. Washington, DC, EUA: IEEE Computer Society, 2013. p. 717–723. ISSN 2160-4886.

SILVA, E. da; ALBINI, L. C. P. Middleware proposals for mobile ad hoc networks. *Journal of Network and Computer Applications*, Elsevier, v. 43, n. 0, p. 103 – 120, 2014. ISSN 1084-8045.

SILVA, E. da; ALBINI, L. C. P.; LIMA, M. W. S. Uma avaliação do esquema de gerenciamento de chave baseado em identidade identity key management. *Revista de Informática Teórica e Aplicada: RITA*, Porto Alegre, RS, Brasil, v. 20, p. 87–101, 2013. ISSN 2175-2745.

SILVA, E. da; e Silva, R. F.; ALBINI, L. C. P. Resisting to false identities attacks to the public-key management system for wireless ad hoc networks. In: *Proceedings of the 3rd ICST Conference on Mobile Networks and Management (MONAMI '11)*. Aveiro, Portugal: ICST, 2011.

SILVA, E. da; LIMA, M. N.; SANTOS, A. L. dos; ALBINI, L. C. P. Identity-based key management in mobile ad hoc networks: Techniques and applications. *IEEE Wireless Communications Magazine*, IEEE Communications Society, Nova Iorque, NY, EUA, v. 15, out. 2008. ISSN 1536-1284.

SILVA, E. da; LIMA, M. N.; SANTOS, A. L. dos; ALBINI, L. C. P. Secure group-based public key management for mobile ad hoc networks. *Journal of Selected Areas in Telecommunications (JSAT)*, Cyber Journals: Multidisciplinary Journals in Science and Technology, v. 2, p. 17–26, 2012. ISSN 1925-2676.

SILVA, E. da; LIMA, M. S.; ALBINI, L. C. P. Demonstrating the security vulnerabilities of the identity-based key management scheme for manets. In: *Proceedings of the 7th IEEE/SBrT International Telecommunications Symposium (ITS'10)*. Manaus, AM, Brasil: IEEE Communications, 2010.

SILVA, E. da; MISAGHI, M.; ALBINI, L. C. P. Distributed self-organized trust management for mobile ad hoc networks. In: *Proceedings of the 4th International Conference on Networked Digital Technologies (NDT '12)*. Dubai, EAU: Springer, 2012. (Communications in Computer and Information Science, v. 293), p. 506–518. ISBN 978-3-642-30506-1.

SILVA, E. da; MISAGHI, M.; ALBINI, L. C. P. True: A trust evaluation service for mobile ad hoc networks resistant to malicious attacks. *Journal of Digital Information Management*, v. 10, n. 4, p. 262–271, 2012. ISSN 0972-7272.

SILVA, E. da; SANTOS, A. L. dos; ALBINI, L. C. P. Gerenciamento de chaves públicas sobrevivente baseado em grupos para manets. In: *Anais do XXX Congresso da Sociedade Brasileira da Computação (CSBC 2010) - Concurso de Teses e Dissertações*. Belo Horizonte, MG, Brasil: SBC, 2010. p. 73–80.

SILVA, E. da; SANTOS, A. L. dos; ALBINI, L. C. P. Gerenciamento de chaves públicas sobrevivente baseado em grupos para manets. In: *Anais do X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SbSeg 2010) - Concurso de Teses e Dissertações*. Fortaleza, CE, Brasil: SBC, 2010. p. 515–522.

SILVA, R. da; KELLERMAN, G. A. Analyzing the payoff of a heterogeneous population in the ultimatum game. *Brazilian Journal of Physics*, scielo, v. 37, p. 1206–1211, dez. 2007. ISSN 0103-9733.

SILVA, R. F. e; SILVA, E. da; ALBINI, L. C. P. A sybil safe virtualization-based public key management scheme for mobile ad hoc networks. *Journal of Selected Areas in Telecommunications (JSAT)*, Cyber Journals: Multidisciplinary Journals in Science and Technology, v. 4, p. 1–8, 2014. ISSN 1925-2676.

SRINIVASAN, V.; NUGGEHALLI, P.; CHIASSERINI, C.; RAO, R. Cooperation in wireless ad hoc networks. In: *Proceedings of the 22th Annual Joint Conference of the IEEE Computer and Communications (INFOCOM 2003)*. San Francisco, CA, EUA: IEEE Societies, 2003. v. 2, p. 808–817. ISSN 0743-166X.

STALLINGS, W. *Criptografia e Segurança de Redes: Princípios e Práticas*. 4. ed. São Paulo, SP, Brasil: Prentice Hall, 2009. ISBN 978-85-7605-119-0.

SU, M.-Y. Prevention of selective black hole attacks on mobile ad hoc networks through intrusion detection systems. *Computer Communications*, v. 34, p. 107–117, 2011. ISSN 0140-3664.

SUN, Y. L.; HAN, Z.; YU, W.; LIU, K. J. R. A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks. In: *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM '06)*. Barcelona, Espanha: IEEE Communications Society, 2006. p. 1–13. ISSN 0743-166X.

SUN, Y. L.; YU, W.; HAN, Z.; LIU, K. J. R. Information theoretic framework of trust modeling and evaluation for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, IEEE Press, v. 24, n. 2, p. 305–317, 2006.

TAMILSELVAN, L.; SANKARANARAYANAN, V. Prevention of blackhole attack in manet. In: *Proceedings of the The 2nd International Conference on Wireless Broadband and Ultra Wideband Communications (AUSWIRELESS '07)*. Washington, DC, EUA: IEEE Computer Society, 2007. p. 21–. ISBN 0-7695-2842-2.

TANEJA, S.; KUSH, A. A survey of routing protocols in mobile ad hoc networks. *International Journal of Innovation, Management and Technology*, v. 1, ago. 2010. ISSN 2010-0248.

TANENBAUM, A. S.; STEEN, M. van. *Sistemas distribuídos: princípios e paradigmas*. 2. ed. São Paulo, SP, Brasil: Prentice Hall, 2007. ISBN 9788576051428.

TATE, J. *wc*-groups over *p*-adic fields. *Séminaire Bourbaki*, Société Mathématique de France, Paris, França, v. 4, p. 265–277, 1956–1958.

THALER, R. H. Anomalies: The ultimatum game. *The Journal of Economic Perspectives*, American Economic Association, v. 2, n. 4, p. 195–206, 1988. ISSN 08953309.

THEODORAKOPOULOS, G.; BARAS, J. S. On trust models and trust evaluation metrics for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, v. 24, n. 2, p. 318–328, 2006.

TIAN, J.; DENKO, M. K. Exploiting clustering and cross-layer design approaches for data caching in MANETs. In: *Proceedings of the 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WIMOB '07)*. Washington, DC, EUA: IEEE Computer Society, 2007. p. 52–59. ISBN 0-7695-2889-9.

VELLOSO, P. B.; LAUFER, R. P.; DUARTE, O.-C.; PUJOLLE, G. A trust model robust to slander attacks in ad hoc networks. In: *Proceedings of 17th International Conference on Computer Communications and Networks. (ICCCN '08)*. Washington, DC, EUA: IEEE Communications Society, 2008. p. 1–6. ISSN 1095-2055.

VERVERIDIS, C. N.; POLYZOS, G. C. Service discovery for mobile ad hoc networks: A survey of issues and techniques. *Communications Surveys & Tutorials*, IEEE Press, Piscataway, NJ, EUA, v. 10, n. 3, p. 30–45, jul. 2008. ISSN 1553-877X.

VIJAYALAKSHMI, S.; RABARA, S. A. Weeding wormhole attack in MANET multicast routing using two novel techniques - $LP^3$ and $NAWA^2$. *International Journal of Computer Applications*, Foundation of Computer Science, Nova Iorque, NY, EUA, v. 16, n. 7, p. 26–33, fev. 2011.

VIRENDRA, M.; JADLIWALA, M.; CH, M.; UPADHYAYA, S. Quantifying trust in mobile ad-hoc networks. In: *Proceedings of the IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS '05)*. Boston, MA, EUA: IEEE Computer Society, 2005. p. 65–71.

WANG, A. I.; BJORNSGARD, T.; SAXLUND, K. Peer2me: Rapid application framework for mobile peer-to-peer applications. In: *Proceedings of the International Symposium on Collaborative Technologies and Systems (CTS '07)*. Orlando, FL, EUA: IEEE Press, 2007. p. 379–388. ISBN 978-0-9785699-1-4.

WEIL, A. Sur les fonctions algébriques à corps de constantes fini. *Les Comptes rendus de l'Académie des sciences*, French Academy of Sciences, Paris, França, v. 210, p. 592–594, 1940.

WU, B.; CHEN, J.; WU, J.; CARDEI, M. A survey on attacks and countermeasures in mobile ad hoc networks. In: _____. *Wireless/Mobile Network Security*. Nova Iorque, NY, EUA: Springer-Verlag, 2006. cap. 12, p. 103–136.

YI, P.; DAI, Z.; ZHANG, S.; ZHONG, Y. A new routing attack in mobile ad hoc networks. *International Journal of Information Technology*, v. 11, p. 83–94, 2005.

YI, S.; KRAVETS, R. MOCA: Mobile certificate authority for wireless ad hoc networks. In: *Proceedings of the 2nd Annual PKI Research Workshop (PKI '03)*. Gaithersburg, MD, EUA: NIST – National Institute of Standards and Technology, 2003.

ZAK, P. J.; KURZBAN, R.; AHMADI, S.; SWERDLOFF, R. S.; PARK, J.; EFREMIDZE, L.; REDWINE, K.; MORGAN, K.; MATZNER, W. Testosterone administration decreases generosity in the ultimatum game. *PLoS ONE*, Public Library of Science, v. 4, n. 12, dez. 2009.

ZAPATA, M. G.; ASOKAN, N. Securing ad hoc routing protocols. In: *Proceedings of the 1st ACM workshop on Wireless security (WiSE '02)*. Nova Iorque, NY, EUA: ACM, 2002. p. 1–10. ISBN 1-58113-585-8.

ZHANG, L.; WU, Q.; QIN, B.; DOMINGO-FERRER, J. Provably secure one-round identity-based authenticated asymmetric group key agreement protocol. *Information Sciences*, Elsevier Science Inc., Nova Iorque, NY, EUA, v. 181, n. 19, p. 4318–4329, out. 2011. ISSN 0020-0255.

ZHANG, X.; NEGLIA, G.; KUROSE, J.; TOWSLEY, D. Performance modeling of epidemic routing. *Computer Networks*, v. 51, n. 10, p. 2867 – 2891, 2007. ISSN 1389-1286.

ZHAO, S.; AGGARWAL, A.; FROST, R.; BAI, X. A survey of applications of identity-based cryptography in mobile ad hoc networks. *IEEE Communications Surveys and Tutorials*, v. 14, n. 2, p. 380–400, 2012.

ZHENG, S.; JIANG, T.; BARAS, J. S. Exploiting trust relations for nash equilibrium efficiency in ad hoc networks. In: *IEEE International Conference on Communications (ICC), 2011*. Kyoto, Japão: [s.n.], 2011. p. 1–5. ISSN 1550-3607.

ZHONG, J. C. S.; YANG, Y. R. Sprite: A simple, cheat-proof, credit-based system for mobile ad hoc networks. In: *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '03)*. San Francisco, CA, EUA: IEEE Communications Society, 2003.

ZHOU, L.; HAAS, Z. J. Securing ad hoc networks. *IEEE Network*, IEEE Communications Society, Los Alamitos, CA, EUA, v. 13, n. 6, p. 24–30, 1999.

ZOURIDAKI, C.; MARK, B. L.; HEJMO, M.; THOMAS, R. K. Robust cooperative trust establishment for manets. In: *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*. Nova Iorque, NY, EUA: ACM, 2006. (SASN '06), p. 23–34. ISBN 1-59593-554-1.

ZUO, Y.; HU, W.-c.; O'KEEFE, T. Trust computing for social networking. In: *Proceedings of the 6th International Conference on Information Technology: New Generations (ITNG '09)*. Washington, DC, EUA: IEEE Computer Society, 2009. p. 1534–1539. ISBN 978-0-7695-3596-8.

# APÊNDICE A

# AMEAÇAS E ESTRATÉGIAS DE DEFESA NAS REDES AD HOC MÓVEIS

Este apêndice apresenta as principais ameaças de segurança contra as MANETs. Contudo, é importante ressaltar que algumas dessas ameaças não são exclusivas das redes *ad hoc*, mas herdadas das redes sem fio em geral, como os ataques de ruído, exaustão e colisão, comuns em qualquer tipo de comunicação sem fio. Além disso, são discutidas estratégias de defesa que podem proteger as redes desses ataques.

## A.1 Ameaças e estratégias de defesa nas camadas física e de enlace

Os ataques mais comuns encontrados nas camadas física e de enlace são consequentes das características da comunicação sem fio. Entre esses ataques, encontram-se a interceptação ou obstrução do sinal, egoísmo e monitoramento e análise dos dados. Várias abordagens foram estudadas e estão sendo aplicadas para mitigar o efeito desses ataques. Tais abordagens serão apresentadas nesta seção, destacando como elas podem ser usadas para proteger as MANETs.

### A.1.1 Interceptação ou obstrução do sinal

Um sinal de rádio, próprio das comunicações sem fio, pode ser facilmente interceptado ou obstruído (EARLE, 2006). Muitas vezes a obstrução ou interferência da comunicação pode ser não-intencional, quando um nó transmite sinais em uma frequência ocupada sem verificar previamente se ela está em uso ou quando vários equipamentos distintos enviam sinais em uma mesma frequência.

A técnica de espalhamento do espectro tem sido amplamente utilizada para permitir

que muitos usuários usem simultaneamente a mesma faixa de frequência sem interferir significativamente um com outro (RAPPAPORT, 2001). Ela também dificulta a ação de atacantes que buscam interceptar o sinal que está sendo transmitido. Nesta técnica, a forma da onda é controlada por uma sequência pseudoaleatória que pode ser determinada facilmente pelos receptores da comunicação, mas que dificulta a captura do sinal transmitido.

As duas principais técnicas de espalhamento do espectro são o *Frequency Hopping Spread Spectrum* (FHSS) ou *Direct Sequence Spread Spectrum* (DSSS). Ambas as técnicas dificultam a intercepção de sinais de rádio, pois um atacante deve conhecer a frequência, o código de espalhamento e a técnica de modulação para ler corretamente um sinal transmitido. A figura A.1 mostra o funcionamento dessas duas técnicas.

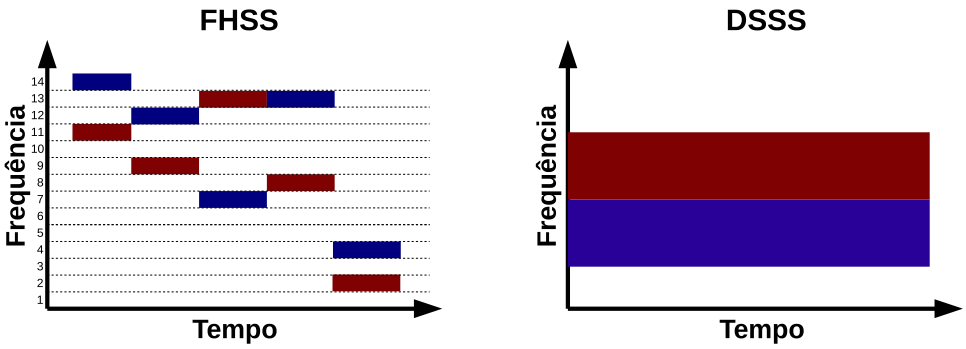



Figura A.1: Espalhamento do espectro com FHSS e DSSS.

O FHSS depende da mudança rápida na frequência de transmissão seguindo um padrão de salto predeterminado e pseudoaleatório. Como ilustrado na figura A.1, o eixo de frequência é divido em uma série *slots*. O padrão de salto controla como os *slots* de transmissão são usados. No exemplo da figura, um padrão de salto é {14,12,7,13,14} e {11,9,13,8,2}. O tempo dos saltos é o elemento chave do FHSS, sendo que o transmissor e o receptor devem estar sincronizados. Já o DSSS aplica um código (ou *chip*) à cadeia de dados, uma sequência de números binários aplicados no processo emissor. No DSSS, o emissor usa sempre a mesma frequência (figura A.1), mas a cadeia de codificação é aplicada aos *bits* de dados.

### A.1.2 Egoísmo

Os protocolos de controle de acesso ao meio das redes sem fio, como o IEEE 802.11, usam mecanismos distribuídos, baseados em cooperação, para compartilhar o canal sem fio de forma justa e eficaz (IEEE, 1999). No protocolo IEEE 802.11 um emissor transmite um *Request To Send* (RTS) após esperar um tempo, chamado de *backoff* aleatório. Assim, um receptor que observa os intervalos entre as transmissões não pode distinguir emissores bem comportados que aleatoriamente obtém um *backoff* pequeno daqueles que selecionam um *backoff* não-aleatório pequeno (KYASANUR; VAIDYA, 2003). Neste contexto, um atacante pode desobedecer a regra de acesso ao meio sem fio com o objetivo de maximizar o seu uso da rede. Esse tipo de atacante é chamado de egoísta, visto que ele tenta comprometer o funcionamento normal da rede em benefício próprio.

Prevenir ou detectar esse tipo de ataque ainda é um desafio para as MANET, contudo vários esquemas têm sido propostos (CáRDENAS; RADOSAVAC; BARAS, 2004; KYASANUR; VAIDYA, 2003; LIU et al., 2009). O algoritmo de detecção de Kyasanur e Vaidya (KYASANUR; VAIDYA, 2003) propõe uma modificação ao IEEE 802.11 para permitir que o receptor identifique emissores malcomportados, por meio de algumas observações. Nesse caso, em vez do emissor selecionar valores de *backoff* aleatórios, é o receptor quem seleciona esses valores e envia nas mensagens de *Clear To Send* (CTS) e pacotes de reconhecimento para o emissor. Esses valores são usados pelo emissor nas suas próximas transmissões ao receptor. Assim, o receptor pode detectar emissores malcomportados, caso os tempos estabelecidos não sejam obedecidos.

O ERA-IEEE 802.11 (CáRDENAS; RADOSAVAC; BARAS, 2004) propõe uma extensão ao IEEE 802.11 para garantir um *backoff* aleatório uniformemente distribuído. O objetivo é que o emissor e o receptor de uma comunicação cheguem a um acordo do valor aleatório por meio de uma discussão pública. Como o ERA-IEEE 802.11 considera que ao menos uma das partes comunicantes é honesta, este nó garante que o valor acordado entre as partes seja realmente aleatório. Para detectar o mau comportamento de algum nó, os autores propõem o uso do algoritmo de detecção de Kyasanur e Vaidya (KYASANUR; VAIDYA, 2003), com algumas modificações para detectar conluio de nós maliciosos. Con-

tudo, esta abordagem exige trocas de mensagens adicionais para que os nós comunicantes cheguem a um acordo, o que implica em uma sobrecarga de comunicação relativamente alta à rede.

O SWN-CUSUM (LIU et al., 2009) é um mecanismo baseado no teste de Soma Cumulativa (PAGE, 1954), identificando estatísticas em tempo real para detectar o comportamento malicioso dos nós. Esta técnica pode ser usada com qualquer protocolo de acesso ao meio, já que não exige qualquer modificação nos protocolos existentes.

### A.1.3 Monitoramento ou análise dos dados

Outra forma de ataque comum na camada de enlace é o monitoramento ou análise dos dados trafegados pela rede. Tais dados podem ser utilizados para obter informações sobre a topologia da rede, entre outros. Esse tipo de ataque afeta a confidencialidade das transmissões e pode ser amenizado com a utilização de algoritmos de cifração de dados. Nas MANETs, que utilizam o padrão IEEE 802.11 para comunicação, têm sido utilizados os protocolos *Wired Equivalent Privacy* (WEP) e *Wi-fi Protected Access* (WPA).

O objetivo dos desenvolvedores do WEP era fornecer o mesmo grau de segurança aos usuários que utilizam as rede com cabos, como a rede Ethernet (CHANDRA, 2005). O protocolo WEP é baseado no algoritmo de cifração RC4[1], que é aplicado aos dados de cada quadro e também ao campo de Verificação de Redundância Cíclica (*Ciclic Redundancy Check* (CRC)). No WEP, a chave de cifração de dados pode ser de 64 ou 128 *bits*, considerando o vetor de inicialização que tem 24 *bits*. Um esquema genérico da cifração dos dados utilizando o WEP é apresentado na Figura A.2.

O protocolo WEP não fornece nenhum mecanismo de estabelecimento de chaves aos seus usuários. Além disso, a segurança da rede é confiada em chaves pré-compartilhadas, que deveriam ser estabelecidas por um mecanismo "fora da banda". O esquema criptográfico do WEP, baseado no RC4, é altamente vulnerável e não garante a confidencialidade das mensagens cifradas.

O protocolo WPA surgiu com o objetivo de resolver os problemas de segurança do

[1]RC4 - Ron's Code versão 4: recebe o nome de seu criado Ron Rivest

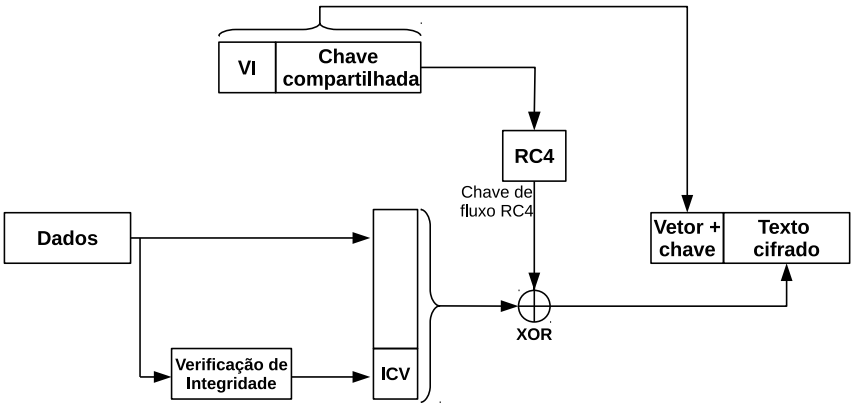



Figura A.2: Esquema genérico do padrão de cifração usando WEP

WEP. Inicialmente, ele foi proposto considerando o uso do Padrão de Cifração Avançado (*Advanced Encryption Standard* (AES)) (DAEMEN; RIJMEN, 2002) como mecanismo de cifração. Porém, por falta de compatibilidade com o *hardware*, foi disponibilizado também o uso do Protocolo de Integridade de Chave Temporal (*Temporal Key Integrate Protocol* (TKIP)). Dessa forma, surgiram dois protocolos distintos: WPA e WPA2. Um esquema genérico da cifração dos dados utilizando o WPA é apresentado na Figura A.3.

Contudo, o WPA com TKIP também apresenta vulnerabilidades pelo uso do algoritmo RC4. Entre estas vulnerabilidades está o ataque de dicionários por força bruta. Dessa forma, para garantir a segurança na cifração de dados em redes sem fio, é aconselhado o uso do WPA2. Um esquema genérico do WPA2 é ilustrado na Figura A.4, em que o algoritmo AES é usado na verificação de integridade dos dados e para cifrar blocos de dados de 128 *bits*.

O WPA e o WPA2 possuem dois métodos de autenticação e gerenciamento de chaves. O primeiro mecanismo de autenticação utiliza o Protocolo de Autenticação Extensível

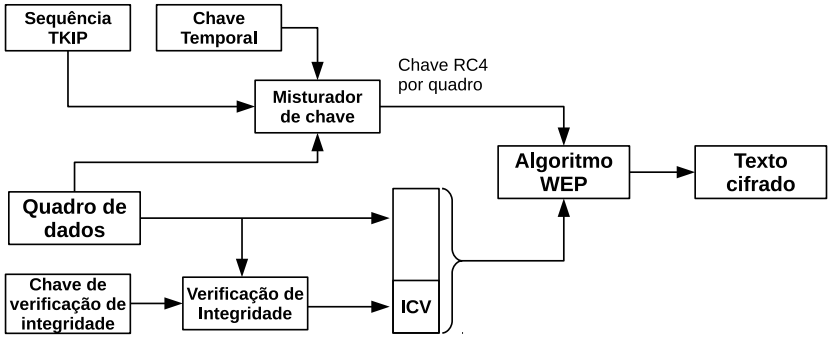



Figura A.3: Esquema genérico do padrão de cifração usando WPA

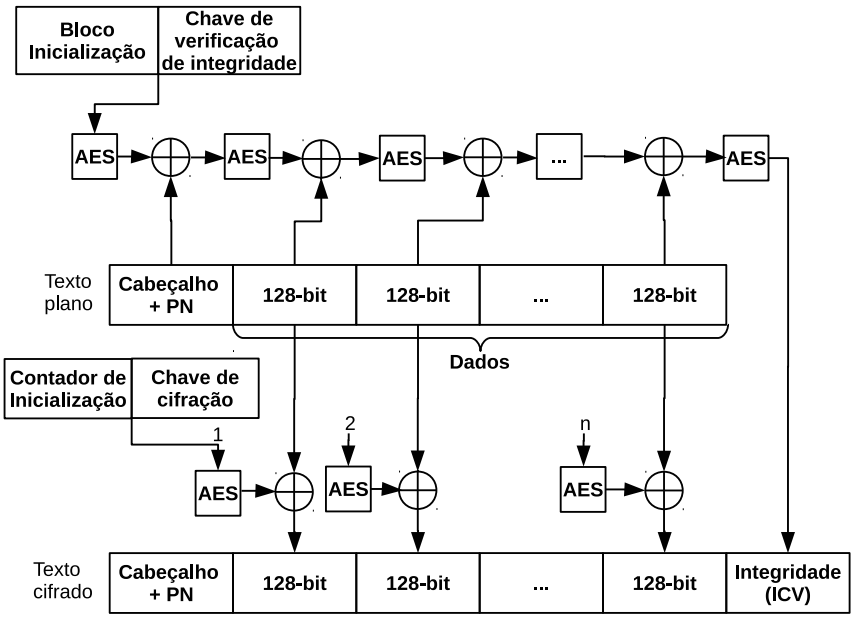



Figura A.4: Esquema genérico do padrão de cifração usando WPA2

(*Extensible Authentication Protocol* (EAP)) (ABOBA et al., 2004), utilizando o padrão IEEE 802.1x (IEEE, 2004) e uma infraestrutura com um servidor de autenticação, como o *Remote Authentication Dial In User Service* (RADIUS) (RIGNEY et al., 2000). O segundo mecanismo usa chaves pré-compartilhadas, que devem ser configuradas em todos os nós.

Para reduzir a exposição da chave mestre, o WPA e o WPA2 adicionam uma camada adicional na hierarquia de chaves, da seguinte forma (CHANDRA, 2005):

a. na primeira se encontra a chave mestre simétrica;

b. na segunda está a chave simétrica transiente derivada da chave mestra simétrica;

c. por fim, está a chave de cifração por pacotes gerada a partir da chave simétrica transiente, por uma função misturadora de chave.

## A.2 Ameaças e estratégias de defesas na camada de rede

O roteamento é considerado um dos principais desafios das MANETs (BOUKERCHE et al., 2011). Devido à topologia dinâmica, as rotas são quebradas com muita facilidade e são imprevisíveis (TANEJA; KUSH, 2010). Por consequência, os protocolos de

roteamento para as MANETs devem ser distribuídos, adaptáveis às frequentes mudanças na topologia da rede e leves (DJENOURI; KHELLADI; BADACHE, 2005). Vários protocolos de roteamento têm sido propostos para MANETs (ALBINI et al., 2006; CLAUSEN; JACQUET, 2003; HAAS; PEARLMAN, 2001; JOHNSON; MALTZ, 1996; PARK; CORSON, 1997; PERKINS; BELDING-ROYER, 1999; PERKINS; BHAGWAT, 1994). Entre esses, o DSR (JOHNSON; MALTZ, 1996) e o *Ad hoc On-Demand Distance Vector* (AODV) (PERKINS; BELDING-ROYER, 1999), são os mais populares e os mais utilizados. Esses dois protocolos são considerados reativos ou sob-demanda pois não mantêm informação sobre a topologia da rede ou rotas entre os nós. Quando um nó deseja enviar uma mensagem, ele deve primeiro descobrir uma rota para o destino, para somente então mandar a mensagem.

A principal diferença entre o DSR e o AODV é forma que eles mantém informações sobre as rotas: no DSR elas são armazenadas na origem enquanto no AODV elas são armazenadas nos nós intermediários. Contudo, a fase de descoberta de rota de ambos é baseada em inundação de pacotes, ou *flooding*. Dessa forma, todos os nós da rede participam em todas as descobertas de rotas. No DSR (JOHNSON; MALTZ, 1996), o nó de origem cria um pacote de Pedido de Rota (*Route Request* (RREQ)) que é enviando para todos os demais nós da rede. Este pacote mantém uma lista dos nós visitados durante a propagação pela rede. Quando o RREQ chega ao destino ou a algum nó que conheça o destino, este nó responde com um pacote de Resposta de Rota (*Route Reply* (RREP)) usando o caminho inverso da rota descoberta pelo RREQ.

Já no AODV (PERKINS; BELDING-ROYER; DAS, 2003; PERKINS; BELDING-ROYER, 1999), os pacotes de RREQ não armazenam a lista dos nós visitados. Neste caso, os nós visitados armazenam informações sobre os nós de origem e destino e o último nó que propagou o RREQ. Assim, os nós intermediários vão mantendo informações para conseguirem chegar à origem novamente. Quando um RREQ chega ao destino ou a um nó que conheça o destino, este nó responde com um RREP usando a rota previamente configurada. Na propagação do RREP à origem, os nós intermediários armazenam também o último nó que enviou o pacote, para configurar as informações de rota ao destino. Um

estudo detalhado sobre outros protocolos de roteamento para as MANETs é apresentado em (BOUKERCHE et al., 2011).

Como as MANETs são altamente dinâmicas, os protocolos de roteamento precisam proporcionar ferramentas para a manutenção das rotas que estiverem quebradas. Geralmente, os protocolos utilizam mensagens de Erro de Rota (*Route Error* (RERR)): quando um nó detecta algum erro em uma rota, ele envia um RERR à origem ou aos nós intermediários da rede, para informar sobre o erro detectado. A forma como a manutenção das rotas funciona pode variar entre os vários protocolos existentes, mas sempre é iniciada por algum nó que detecta a anomalia nas rotas (JOHNSON; MALTZ, 1996; PERKINS; BELDING-ROYER, 1999).

Outra caracterísitica dos protocolos de rotamento para as MANETs é que eles dependem da cooperação efetiva dos nós e, geralmente, assumem que os nós possuem um bom comportamento e são confiáveis. Contudo, caso os nós apresentem um comportamento malicioso, diversos tipos de ataques podem ser realizados, comprometendo as operações de roteamento (AGRAWAL; JAIN; SHARMA, 2011). Esses ataques podem ser classificados em duas grandes categorias (HU; PERRIG, 2004): ataques de rompimento de rotas e ataques de consumo de recursos. No primeiro, um atacante tem como objetivo rotear os pacotes da rede por rotas falsas e não funcionais. Já no segundo tipo de ataque, seu objetivo é injetar pacotes na rede, consumindo recursos valiosos como banda, energia, processamento e outros.

A seguir são apresentados os tipos de ataques mais comuns que são realizados contra a camada de redes nas MANETs: inundação, falta de cooperação, buraco negro, buraco de minhoca, aceleração, modificação e fabricação. Além disso, são apresentadas soluções propostas para mitigar o impacto desses ataques. Alguns estudos mais detalhados sobre as soluções de segurança para os protocolos de roteamento nas MANETs podem ser encontrados na literatura (AGRAWAL; JAIN; SHARMA, 2011; HU; PERRIG, 2004; HU; PERRIG; JOHNSON, 2005; KANNHAVONG et al., 2007; SANZGIRI et al., 2002; HU; PERRIG; JOHNSON, 2003a; ZAPATA; ASOKAN, 2002).

### A.2.1 Inundação

Em um ataque de inundação, um atacante envia muitos pacotes de criação de rotas para nós inexistentes, avisos de rotas em excesso ou pedidos de rota (AGRAWAL; JAIN; SHARMA, 2011). Os objetivos de um ataque de inundação são: congestionar os enlaces sem fio e esgotar os recursos da rede. É um exemplo clássico de um ataque de consumo de recursos.

Vários protocolos de roteamento foram desenvolvidos com o objetivo de mitigar o efeitos dos ataques de inundação no roteamento das MANETs. Por exemplo, Yi et al. (2005) propuseram uma extensão ao protocolo AODV (PERKINS; BELDING-ROYER; DAS, 2003), chamada de *Flooding Attack Prevention* (FAP) (YI et al., 2005), em que cada nó monitora os pedidos de rotas de seus vizinhos. Caso os pedidos de rota de um único nó ultrapassem um limite preestabelecido, esse nó é adicionado a uma lista negra local. Assim, cada nó possui uma lista negra contendo possíveis nós maliciosos e pode descartar os pedidos de rotas desses nós.

Uma variação do FAP propõe o uso de um mecanismo adaptativo para descarte estatístico de pacotes (DESILVA; BOPPANA, 2005). Neste caso, o descarte dos pacotes não é baseado em um limite fixo, mas em uma análise estatística dos pedidos de rota. Essa abordagem reduz o impacto de ataques com variação da taxa de inundação.

O Ariadne (HU; PERRIG; JOHNSON, 2005) é um outro protocolo que previne, entre outros, ataques de inundação. Ele utiliza primitivas da criptografia simétrica para proteger o roteamento dos pacotes. Além disso, para resistir a ataques de inundação, ele autentica as mensagens de pedidos de rotas e define limites desses pedidos. Dessa forma, se um nó realizar muitos pedidos de rotas, os demais nós podem descartar esses pedidos, assumindo que este nó esteja realizando um ataque.

### A.2.2 Falta de cooperação

Um nó também pode apresentar um mau comportamento quando utiliza os recursos de encaminhamento de pacotes, mas se recusa a cooperar nessas atividades (MARTI et al., 2000). Embora os nós possam agir maliciosamente, geralmente eles apresentam esse

mau comportamento pois desejam economizar os seus recursos, como energia, memória ou processamento (MARIAS et al., 2006). Esse tipo de ataque pode afetar a disponibilidade e a robustez da rede, além de diminuir a eficácia e a vazão da comunicação.

As principais estratégias para prevenir esse tipo de ataque consideram o uso de técnicas de incentivo à cooperação para encorajar a colaboração entre os nós. Essas técnicas podem ser baseadas na reputação dos nós (BUCHEGGER; BOUDEC, 2002a; BUCHEGGER; BOUDEC, 2002b; MICHIARDI; MOLVA, 2002) ou baseadas em crédito (BUTTYÁN; HUBAUX, 2001; ZHONG; YANG, 2003).

Os esquemas baseados em reputação (BUCHEGGER; BOUDEC, 2002a; BUCHEGGER; BOUDEC, 2002b; MICHIARDI; MOLVA, 2002) utilizam como métrica a reputação dos nós para o encaminhamento de mensagens. Essa reputação é geralmente medida por nós considerados confiáveis. A reputação de um nó aumenta à medida que ele executa as tarefas de encaminhamento de mensagens corretamente. De forma geral, tais esquemas possuem técnicas para isolar os nós maliciosos, que devem possuir baixa reputação.

As técnicas baseadas em reputação podem tomar decisões considerando as observações diretas realizadas pelos nós, bem como as recomendações de outros nós. Para isso, os nós trocam informações sobre a reputação dos outros. Entre os algoritmos de incentivo à cooperação, encontram-se o Confidant (BUCHEGGER; BOUDEC, 2002a; BUCHEGGER; BOUDEC, 2002b) e o Core (MICHIARDI; MOLVA, 2002). Esses dois algoritmos estimulam a cooperação dos nós unindo um monitoramento colaborativo e um mecanismo de reputação. Cada nó monitora o comportamento de seus vizinhos e diminui o valor da reputação de todos os nós que se recusam a cooperar no encaminhamento de pacotes. Caso esse comportamento de não-cooperação persista, então esse nó malicioso será excluído da rede.

As técnicas baseadas em crédito (BUTTYÁN; HUBAUX, 2001; ZHONG; YANG, 2003) assumem que a tarefa de encaminhamento de pacotes é um serviço que pode ser comercializado, no qual valores podem ser aplicados e cobrados para a realização dela. Essas técnicas requerem o uso de algum tipo de moeda virtual para regular as negociações entre os nós e necessitam ou de um *hardware* resistente a alterações ou de um banco

virtual (MARIAS et al., 2006).

Nos esquemas que utilizam um *hardware* resistente a alterações, todos os nós devem possuir esse *hardware* específico, o que pode encarecer a implementação do algoritmo. Por outro lado, os esquemas que utilizam banco virtual precisam de uma entidade confiável para oferecer esse serviço, o que nem sempre é desejável nas MANETs. Entre os diversos algoritmos baseados em créditos estão o Nuglets (BUTTYÁN; HUBAUX, 2001), que considera o uso de um *hardware* resistente a alterações, e o Sprite (ZHONG; YANG, 2003) que utiliza a abordagem de banco virtual.

### A.2.3 Buraco Negro

Em um ataque "buraco negro", ou *blackhole*, todos os pacotes que chegam a um nó malicioso para serem roteados são descartados (DENG; LI; AGRAWAL, 2002). Geralmente durante a fase de descoberta das rotas, os nós *blackhole* apresentam um bom comportamento, cooperando com as atividades da rede e ganhando a confiança dos demais nós. Ao serem escolhidos para participar do roteamento dos pacotes, eles passam a se comportar maliciosamente (AGRAWAL; GHOSH; DAS, 2008). Dessa forma, se um nó malicioso participar de muitas rotas, esse tipo de ataque pode levar ao particionamento da rede. Uma variação desse ataque, chamada de *selective forwarding* ou *grayhole*, descarta os pacotes de dados seletivamente, podendo levar à perda de desempenho da rede ou até ao isolamento de alguns nós (HU; PERRIG, 2004). Essa variação do *blackhole* é ainda mais difícil de ser detectada.

Muitos protocolos de roteamento seguro para MANETs têm como objetivo serem resistentes a esses ataques. Em (TAMILSELVAN; SANKARANARAYANAN, 2007) é proposto um algoritmo para estender as funcionalidades do AODV. Nele, um nó não envia pacotes de dados ao nó intermediário, que respondeu ao pedido. Pelo contrário, ele aguarda por outras respostas de seus vizinhos com detalhes sobre o próximo salto. Após receber as respostas, ele procura por informações que confirmem o próximo salto, para então confirmar que esse caminho é correto.

Outra solução é o *Detection, Prevention and Reactive AODV* (DPRAODV), que isola

nós maliciosos da rede (RAJ; SWADAS, 2009). Esse protocolo emprega um sistema de aprendizado dinâmico (KUROSAWA et al., 2007) para detectar nós que estejam realizando um ataque *blackhole*. Essa abordagem pode detectar um ataque com um baixo custo, pois não requer trocas de mensagens extras. Uma outra abordagem é o *Anti-Blackhole Mechanism* (ABM) (SU, 2011) que estima a possibilidade de um nó ser malicioso baseada em diferenças anormais entre mensagens de roteamento transmitidas a partir desse nó. Para realizar essa função, vários nós com a função de *Intrusion Detection System* (IDS) são implantados na rede para observarem os pedidos de rotas dos nós e o número de encaminhamento desses pedidos, a fim de julgar se algum nó está agindo maliciosamente.

### A.2.4 Buraco de Minhoca

O "buraco de minhoca", ou *wormhole*, é um tipo de ataque em que dois nós, geograficamente distantes, criam um "túnel" e direcionam os pacotes por ele (HU; PERRIG; JOHNSON, 2005). Esse túnel pode ser criado usando um canal paralelo de baixa latência ou por meio de encapsulamento de pacotes (HU; PERRIG; JOHNSON, 2003a). Os pacotes transmitidos por túneis *wormholes* podem manter as características originais do ponto em que foi recebido. Isso pode comprometer o conhecimento que os nós têm da sua vizinhança. O principal objetivo desse ataque é ampliar o efeito de outros ataques, como o Buraco Negro ou o monitoramento de pacotes, já que o canal estabelecido pelo ataque consegue privilegiar sua participação no roteamento (HU; PERRIG; JOHNSON, 2006).

Diversas soluções são propostas com o objetivo de defender o roteamento nas MANETs dos ataques *wormhole* (BRUSCHI; ROSTI, 2002; GORLATOVA et al., 2006; HU; PERRIG; JOHNSON, 2006; QIAN; SONG; LI, 2007; VIJAYALAKSHMI; RABARA, 2011). Em (HU; PERRIG; JOHNSON, 2006) é apresentado o *packet leashes*, ou "cerca de pacotes", com cercas temporais e geográficas. As cercas temporais são usadas para evitar que os pacotes viajem por distâncias muito longas, contudo exigem que os nós tenham relógios sincronizados. Nas cercas geográficas, os nós enviam a sua posição geográfica junto aos pacotes. Dessa forma, um receptor pode verificar se um nó é, de fato, seu vizinho, baseado em sua própria posição geográfica.

Outra solução emprega a técnica *Limiting Packet Propagation Parameter* ($LP^3$) (VIJAYALAKSHMI; RABARA, 2011) que adiciona um campo aos pacotes de roteamento, similar ao *Time-To-Live* (TTL). Este campo é um valor aleatório que limita a longa viagem do pacote pelos túneis *wormholes*. Ele é definido de forma que o pacote possa chegar ao seu destino antes da sua expiração. Contudo, para evitar que nós atacantes possam alterar o valor desse campo, ele deve ser cifrado e assinado pelo emissor.

### A.2.5 Aceleração

Com o objetivo de não sobrecarregar a rede, em alguns protocolos de roteamento para as MANETs, cada nó intermediário encaminha apenas os primeiros pacotes de pedido da rota e os demais são descartados (JOHNSON; MALTZ, 1996). Nesse caso, os nós maliciosos podem explorar essa característica e reenviar esses pacotes rapidamente por toda a rede. Como resultado, os próximos pedidos de construção de rotas são descartados pelos demais nós, pois eles acreditam que são pedidos duplicados. Isso aumenta a probabilidade de que as rotas descobertas incluam esse nó malicioso (ANJUM; MOUCHTARIS, 2007). Depois de conseguir participar de várias rotas, um atacante pode realizar outros tipos de ataques, com o objetivo de particionar a rede, reduzir o seu desempenho ou extrair informações relevantes.

Em (HU; PERRIG; JOHNSON, 2003b) são discutidas algumas técnicas para defender o roteamento de ataques de aceleração, que incluem: detecção de nós vizinhos, delegação de rota e encaminhamento aleatório de pedidos de rotas. Inicialmente, um nó $i$ verifica se um outro nó $j$ está dentro do seu raio de transmissão. Uma vez que o nó $i$ detecta que o nó $j$ é seu vizinho ele delega esse nó para encaminhar os seus pedidos de rotas, por meio de uma mensagem assinada de delegação de rota.

Quando o nó $j$ determina que o nó $i$ está dentro de seu raio de comunicação, ele aceita a delegação de rota, por meio de uma mensagem assinada de aceite de delegação. Por fim, ele faz uma seleção aleatória das mensagens de pedido de rota, que garante que os nós que encaminham os pedidos com baixa latência são selecionados com uma probabilidade apenas ligeiramente maior que os demais caminhos.

### A.2.6 Modificação e fabricação

Os ataques de modificação e fabricação são comuns nas MANETs, visto que os próprios nós cooperam na criação das rotas (AGRAWAL; JAIN; SHARMA, 2011). Dessa forma, um atacante pode modificar os caminhos criados, fabricar rotas falsas ou emitir mensagens falsas de erro de rotas. Esse tipo de ataque pode levar à alteração da topologia da rede e à perda de desempenho (BANERJEE; SWAMINATHAN, 2011).

Geralmente, as soluções para prevenir esse tipo de ataque usam algum tipo de criptografia. O Ariadne (HU; PERRIG; JOHNSON, 2005), por exemplo, tem como objetivo fornecer um mecanismo de autenticação com baixo custo computacional e baixa sobrecarga de comunicação. Ele garante, na fase de descoberta de rota, que o nó destino possa autenticar o nó inicial e o nó inicial possa autenticar cada nó intermediário do caminho. Além disso, ele realiza a autenticação das mensagens tanto na fase de construção das rotas como na fase de manutenção. Com isso, ele previne mensagens falsas de erro de rota e a criação de rotas com nós malcomportados (MOLVA; MICHIARDI, 2003).

Outro exemplo é o *Authenticated Routing for Ad hoc Networks* (ARAN) (SANZGIRI et al., 2002), no qual cada nó tem um certificado assinado por uma autoridade confiável $T$. Todos os nós da rede possuem acesso à chave pública de $T$ e, assim, eles podem verificar a autenticidade dos certificados (ANJUM; MOUCHTARIS, 2007). Outra solução é o *Secure* AODV (SAODV) (ZAPATA; ASOKAN, 2002), que também utiliza assinaturas digitais para autenticar a maioria dos campos das mensagens de requisição e resposta de rotas. Além disso, ele usa cadeias *hash* para autenticar a contagem de saltos. Essas soluções introduzem autenticação, integridade e não-repúdio de mensagens, impedindo a modificação ou fabricação de mensagens falsas.

## A.3 Ameaças às camadas superiores

Diversos ataques também podem afetar as camadas superiores em uma MANET ou podem ser considerados multi-camadas. Nesta seção são apresentadas as principais ameaças encontradas, porém não são apresentadas estratégias para mitigar ou prevenir esses

ataques. No capítulo 3 é apresentado o SEMAN e são discutidas as técnicas que podem ser utilizadas para prevenir tais ataques.

Os ataques mais comuns encontrados na camada de aplicação são (WU et al., 2006): vírus e *worms* móveis, personificação, negação de serviço e Sybil. Os vírus e *worms* móveis são injetados na rede por meio de códigos maliciosos que aproveitam a vulnerabilidade dos protocolos e aplicações. Em um ataque de personificação, um atacante personifica um usuário autorizado, obtendo acesso aos recursos que estão protegidos por autenticação (WU et al., 2006). Uma das formas mais comuns desse tipo de ataque é conhecida como ataque homem-no-meio (*man-in-the-middle*). Nesse tipo de ataque, o atacante se posiciona entre o cliente e a rede roubando informações de autenticação que foram enviadas pelo cliente (CHEN et al., 2007). Em seguida, ele pode usar essas informações para personificar um nó autêntico.

O ataque de negação de serviço pode acontecer de diversas formas, como por exemplo, pelo excesso de envio de dados para a rede com o objetivo de deixá-la congestionada (WU et al., 2006). Por fim, em um ataque Sybil, um atacante assume diversas identidades na rede enquanto utiliza um único dispositivo físico (DOUCEUR, 2001). Esse tipo de ataque pode ter impacto, por exemplo, em algoritmos de roteamentos multi-caminhos ou em mecanismos baseados em confiança ou eleição. Esses sistemas devem garantir que as identidades estejam relacionadas com entidades distintas. As identidades adicionais de um nó Sybil podem ser obtidas de duas formas: o atacante pode fabricar uma nova identidade ou pode roubar uma identidade de um outro nó legítimo (NEWSOME et al., 2004).

## A.4 Conclusão

Este capítulo apresentou as ameaças de segurança encontradas nas MANETs, considerando as camadas física, de enlace de dados e de rede. Também foram discutidas as principais estratégias de defesas propostas para minimizar o impacto dessas ameaças. Não foram encontradas vulnerabilidades na camada de transporte que são consequentes das características das MANETs e, dessa forma, esta camada não foi discutida neste capítulo.

Por fim, foram discutidos os principais ataques que podem ser realizados contra a camada de aplicação nas MANETs, como personificação, negação de serviço e Sybil. Esses ataques podem ser realizados como consequência das vulnerabilidades que as características dinâmicas e auto-organizadas que as aplicações sobre as MANETs possuem. Contudo, as estratégias de defesa para tais vulnerabilidades não foram apresentada neste capítulo. No capítulo 3 é apresentado o SEMAN, que tem como objetivo proteger as aplicações que dessas ameaças.

A Tabela A.1 resume as ameaças estudadas e quais as estratégias de defesa que são utilizadas para amenizar os seus impactos nas MANETs.

Tabela A.1: Ameaças de segurança nas MANETs e estratégias de defesa

| **Camada** | **Ataque** | **Descrição** | **Estratégias de defesa** |
|---|---|---|---|
| Física e Enlace | Interceptação ou obstrução | captura ou interferência no sinal transmitido | Espalhamento do espectro FHSS ou DSSS (RAPPAPORT, 2001) |
| | Egoísmo | atacante desobedece regras de acesso ao meio, para maximizar o seu uso da rede | algoritmos de Kyasanur e Vaidya (KYASANUR; VAIDYA, 2003), ERA-IEEE 802.11 (CáRDENAS; RADOSAVAC; BARAS, 2004) e SWN-CUSUM (LIU et al., 2009) |
| | Monitoramento ou análise | análise dos dados transmitidos | cifração dos dados usando WEP, WPA ou WPA2 |
| Rede | Inundação | atacante inunda a rede com pacotes de criação de rotas para nós inexistentes, avisos de rotas em excesso ou pedidos de rota | algoritmos FAP (YI et al., 2005), de Desilva e Boppana (DESILVA; BOPPANA, 2005) e Ariadne (HU; PERRIG; JOHNSON, 2005) |
| | Falta de cooperação | atacante utiliza os recursos de encaminhamento de pacotes, mas se recusa a cooperar nessa atividade | técnicas de incentivo a cooperação, baseadas na reputação dos nós (BUCHEGGER; BOUDEC, 2002a; BUCHEGGER; BOUDEC, 2002b; MICHIARDI; MOLVA, 2002) ou baseadas em crédito (BUTTYÁN; HUBAUX, 2001; ZHONG; YANG, 2003) |
| | Buraco Negro | nó malicioso exclui todos os pacotes que chegam a ele para ser roteados | algoritmo de Tamilselvan e Sankaranarayanan (TAMILSELVAN; SANKARANARAYANAN, 2007), DPRA-ODV (RAJ; SWADAS, 2009) e ABM (SU, 2011) |
| | Buraco de Minhoca | dois nós criam um túnel para direcionar pacotes | Cercas de pacotes (*packet leashes*) (HU; PERRIG; JOHNSON, 2006) e algoritmo LP$^3$ (VIJAYALAKSHMI; RABARA, 2011), além de (BRUSCHI; ROSTI, 2002; GORLATOVA et al., 2006; QIAN; SONG; LI, 2007). |
| | Aceleração | atacante encaminha rapidamente as mensagens de pedido de rota quando é iniciada uma descoberta de rota | Algoritmo de Hu, Perrig e Johson (HU; PERRIG; JOHNSON, 2003b) |
| | Modificação e fabricação | atacante modifica os caminhos criados, fabrica rotas falsas ou emite mensagens falsas de erros de rotas | algoritmos de roteamento com criptografia, como Aridne (HU; PERRIG; JOHNSON, 2005), ARAN (SANZGIRI et al., 2002), SAODV (ZAPATA; ASOKAN, 2002) |

# APÊNDICE B

# CRIPTOGRAFIA BASEADA EM IDENTIDADE

Em 1984, Adi Shamir apresentou um novo modelo de criptografia assimétrica, chamado de IBC (SHAMIR, 1985), como alternativa para simplificar o gerenciamento de chaves públicas e certificados em uma PKI. Um IBC permite que qualquer par de usuários se comuniquem, de forma segura, e verifiquem mutuamente suas assinaturas sem a troca de chaves públicas e privadas, sem manter um diretório de chaves e sem usar os serviços de uma terceira entidade (ZHAO et al., 2012). Assim, o IBC visa a evitar o alto custo do gerenciamento de chaves públicas e autenticação de assinaturas presente em uma PKI tradicional. Contudo, a proposta de Shamir não apresentou soluções práticas para fornecer um esquema de IBE. Apenas em 2001, Boneh e Franklim (BONEH; FRANKLIN, 2001) apresentaram o primeiro esquema IBE prático e seguro usando mapas bilineares. Esse esquema é conhecido como BF-IBE. Após esse estudo, outros esquemas baseados em identidade foram propostos, como o IBE hierárquico, IBS, autenticação baseada em identidade e protocolos de acordo de chaves. Sem a perda da generalidade, esta tese parte do esquema BF-IBE, embora outro esquema possa ser empregado na construção dos algoritmos.

Em um IBC, em vez de gerar um par aleatório de chaves pública e privada, um usuário escolhe uma *string* arbitrária, como o seu e-mail ou endereço de IP, para ser a sua chave pública. Assim, esse modelo de sistema elimina a necessidade de certificados de chave pública e da propagação dos certificados e chaves públicas dos usuários pela rede. Então, um emissor pode cifrar uma mensagem para um receptor conhecendo apenas a identidade do receptor, sem precisar de um certificado de chave pública. Por outro lado, um usuário não pode emitir a sua própria chave privada. Para isso, é necessária uma entidade confiável para emitir as chaves privadas dos usuários, chamada de PKG. Este PKG é responsável

também pela configuração do sistema e pela geração da chave mestre da rede.

A Figura B.1 apresenta uma exemplo do funcionamento de um IBC. Nesse exemplo, Beto envia uma mensagem para Ana. Ele utiliza a identidade conhecida de Ana associada à chave pública mestre do sistema para cifrar a mensagem que é transmitida. A chave de decifração é solicitada por Ana e gerada pelo PKG. Como um PKG emite todas as chaves privadas dos usuários, ele pode decifrar todas as mensagens desse usuário. Isso acontece porque o PKG detém a chave privada mestre. Esse problema é conhecido como custódia da chave[1]. Assim, os IBCs requerem que o PKG seja totalmente confiável, o que dificulta a sua implementação em ambientes dinâmicos, como as MANETs.

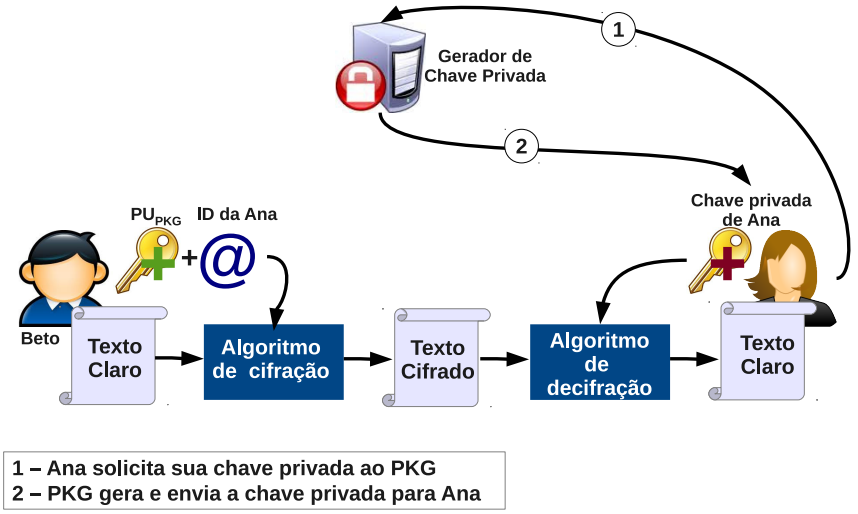



Figura B.1: Visão geral do funcionamento dos criptossistemas baseados em identidade

De modo geral, os esquemas criptográficos baseados em identidade consideram quatro algoritmos: configuração, extração, cifração e decifração. Uma breve descrição de cada um desses algoritmos é apresentada a seguir:

a. "configuração": mapeia *strings* arbitrárias (identidade) para pontos em uma curva elíptica. Configura a chave pública do sistema $PU_{PKG}$ como $sP$, em que $s$ é um número aleatório em $\mathbb{Z}_q^*$ e $P$ é um ponto arbitrário em $E/\mathbb{F}_p$ de ordem $q$. Escolhe uma função *hash* $H : \mathbb{F}_{p^2} \rightarrow \{0,1\}^n$ para algum $n$. Escolhe uma segunda função *hash* $G : \{0,1\}^* \rightarrow \mathbb{F}_p$. Os parâmetros do sistema são publicados como $\langle p, n, P, PU_{PKG}, G, H\rangle$. A chave mestre privada é $s \in \mathbb{Z}_q$;

[1]Comumente encontrado na literatura como *key escrow*.

b. “extração”: para uma dada *string* $ID \in \{0,1\}^*$, o algoritmo constrói a chave pública para $ID$ : $Q.ID = G(ID)$, um ponto em $E/\mathbb{F}_q$ mapeado a partir de $ID$, e chave privada $d.ID = s.Q.ID$;

c. “cifração”: escolhe aleatoriamente $r \in \mathbb{Z}_q$ e gera um texto cifrado $C = rP, M \oplus H(g.ID)$ em que $g.ID = \hat{e}(Q.ID, PU_{PKG}) \in \mathbb{F}_{p^2}$; e

d. “decifração”: Sendo $C = \langle U, V \rangle$ um texto cifrado usando a chave pública de $ID$, o algoritmo decifra $C$ usando a chave privada $d.ID : V \oplus H(\hat{e}(d.ID, U)) = M$.